IN THE UNITED STATES DISTRICT COURT

FOR THE NORTHERN DISTRICT OF CALIFORNIA

SUN MICROSYSTEMS INC,

Plaintiff,

v.

NETWORK APPLIANCE,

Defendant.
_______________________________________/

No. C-08-01641 EDL

ORDER CONSTRUING CLAIM TERMS

On April 1, 2009, the Court held a claim construction hearing to construe the disputed terms of United States Patent Numbers 6,965,951 (the "'951 patent"), 6,151,683 ("'683 patent"), 6,283,249 ("'249 patent"), and 6,484,200 ("'200 patent") (collectively, "Sun patents") pursuant to Markman v. Westview Instruments, Inc., 517 U.S. 370 (1996). On April 15, 2009, the Court held a claim construction hearing to construe the disputed terms of United States Patent Numbers 7,328,305 (the "'305 patent"), 7,293,152 (the "'152 patent"), 6,516,351 ("'351 patent"), and 7,293,097 ("'249 patent") (collectively, "NetApp patents"). Having read the papers and considered the arguments of counsel and the relevant legal authority, the Court hereby rules as follows.

I. BACKGROUND

On March 26, 2008, Network Appliance Inc. ("NetApp") filed its complaint against Sun Microsystems Inc. ("Sun") for patent infringement under 35 U.S.C. § 271. Sun filed its answer and counterclaims on May 19, 2008, to which NetApp filed a reply on June 12, 2008. The parties now seek construction of four disputed terms contained in the Sun patents. The parties also seek construction of three disputed terms contained in the NetApp patents. The parties originally sought construction of two additional terms, but agreed to a stipulated construction of those terms, which the Court adopts as discussed below.

II. LEGAL STANDARD

In construing claims, the court must begin with an examination of the claim language itself. The terms used in the claims are generally given their "ordinary and customary meaning." See Phillips v. AWH Corp., 415 F.3d 1303, 1312-13 (Fed. Cir. 2005); see also Renishaw PLC v. Marposs Societa' per Azioni, 158 F.3d 1243, 1248 (Fed. Cir. 1998) ("[T]he claims define the scope of the right to exclude; the claim construction inquiry, therefore, begins and ends in all cases with the actual words of the claim."). This ordinary and customary meaning "is the meaning that the terms would have to a person of ordinary skill in the art in question at the time of the invention . . . ." Phillips, 415 F.3d at 1313. A patentee is presumed to have intended the ordinary meaning of a claim term in the absence of an express intent to the contrary. York Products, Inc. v. Central Tractor Farm & Family Ctr., 99 F.3d 1568, 1572 (Fed. Cir. 1996).

Generally speaking, the words in a claim are to be interpreted "in light of the intrinsic evidence of record, including the written description, the drawings, and the prosecution history, if in evidence." Teleflex, Inc. v. Ficosa North Am. Corp., 299 F.3d 1313, 1324-25 (Fed. Cir. 2002); see also Medrad, Inc. v. MRI Devices Corp., 401 F.3d 1313, 1319 (Fed. Cir. 2005) (court looks at "the ordinary meaning in the context of the written description and the prosecution history") (citations omitted). "Such intrinsic evidence is the most significant source of the legally operative meaning of disputed claim language." Vitronics Corp. v. Conceptronic, Inc., 90 F.3d 1576, 1582 (Fed. Cir. 1996).

With regard to the intrinsic evidence, the court's examination begins with the claim language. See id. Specifically, "the context in which a term is used in the asserted claim can be highly instructive." Phillips, 415 F.3d at 1314. As part of that context, the court may also consider the other patent claims, both asserted and unasserted. Id. For example, as claim terms are normally used consistently throughout a patent, the usage of a term in one claim may illuminate the meaning of the same term in other claims. Id. The court may also consider differences between claims as a guide to understanding the meaning of particular claim terms. Id.

Second, the claims "must [also] be read in view of the specification, of which they are a part." Id. at 1315 (citation omitted). When the specification reveals a special definition given to a

claim term by the patentee that differs from the meaning it would otherwise possess, the inventor's lexicography governs. Id. at 1316. Indeed, the specification is to be viewed as the "best source" for understanding a technical term, informed as needed by the prosecution history. Id. at 1315. As the Federal Circuit stated in Phillips, the specification is "the single best guide to the meaning of a disputed term," and "acts as a dictionary when it expressly defines terms used in the claims or when it defines terms by implication." 415 F.3d at 1321 (citation omitted).

Limitations from the specification, however, such as from the preferred embodiment, cannot be read into the claims absent a clear intention by the patentee to do so. Altiris, Inc. v. Symantec Corp., 318 F.3d 1363, 1372 (Fed. Cir. 2003) ("resort to the rest of the specification to define a claim term is only appropriate in limited circumstances"); Teleflex, 299 F.3d at 1326 ("[T]he claims must be read in view of the specification, but limitations from the specification are not to be read into the claims.") (citations omitted); CCS Fitness, Inc. v. Brunswick Corp., 288 F.3d 1359, 1366 (Fed. Cir. 2002) ("a patentee need not describe in the specification every conceivable and possible future embodiment of his invention") (internal quotations omitted).

"[T]here is sometimes a fine line between reading a claim in light of the specification, and reading a limitation into the claim from the specification. . . . [A]ttempting to resolve that problem in the context of the particular patent is likely to capture the scope of the actual invention more accurately than either strictly limiting the scope of the claims to the embodiments disclosed in the specification or divorcing the claim language from the specification." Decisioning.com, Inc. v. Federated Dept. Stores, Inc., 527 F.3d 1300, 1307-08 (Fed. Cir. 2008) (quoting Comark Commc'ns, Inc. v. Harris Corp., 156 F.3d 1182, 1186 (Fed. Cir. 1998)). There is therefore "no magic formula or catechism for conducting claim construction," and the court must "read the specification in light of its purposes in order to determine whether the patentee is setting out specific examples of the invention to accomplish those goals, or whether the patentee instead intends for the claims and the embodiments in the specification to be strictly coextensive." Id. at 1308 (internal citations omitted).

Finally, as part of the intrinsic evidence analysis, the court "should also consider the patent's prosecution history, if it is in evidence." Phillips, 415 F.3d at 1317. The court should take into account, however, that the prosecution history "often lacks the clarity of the specification" and thus

is of limited use for claim construction purposes. Id.

In most cases, claims can be resolved based on intrinsic evidence. See Vitronics, 90 F.3d at 1583. Only if an analysis of the intrinsic evidence fails to resolve any ambiguity in the claim language may the court then rely on extrinsic evidence, such as expert and inventor testimony, dictionaries, and learned treatises. See id. ("In those cases where the public record unambiguously describes the scope of the patented invention, reliance on any extrinsic evidence is improper"). "Within the class of extrinsic evidence, the court has observed that dictionaries and treatises can be useful in claim construction." Phillips, 415 F.3d at 1318. While expert testimony can be useful to a court for a variety of purposes, conclusory assertions by experts are not useful to a court. Id. The court generally views extrinsic evidence as less reliable than the patent and its prosecution history in determining how to read claim terms, even though its consideration is within the court's sound discretion. See id. at 1318-19.

III. DISCUSSION

The parties dispute four terms contained in three of the Sun patents: (1) "token" in the '683 and '249 patents; (2) "computer system" in the '683 and '249 patents; (3) "modifiable tree structure including elements in a fixed hierarchical relationship"/"modifiable static tree structure" in the '249 patent; and (4) "discovery interface" in the '951 patent. The parties dispute three terms contained in four of the NetApp patents: (1) "in a non-fixed pattern" in the '305 patent; (2) "initiator group (igroup)" in the '152 patent; and (3) "uniform file-locking semantics"/"uniform locking semantics" in the '351 and '097 patents. The parties agreed to a construction of the term "to examine [examining] the event communication" in the '200 Sun patent and the term "opportunistic locks" in the '351 and '097 NetApp patents.

A. '683 and '249 Patents

The '683 patent is entitled "rebuilding computer states remotely." It relates to "monitoring of computer systems and more particularly to rebuilding the state of a computer system based on diagnostic data from the computer system." '683 patent at 1:17-20. Computer systems have hardware and software components that fail and degrade system performance. Id. at 1:22-25. The '683 patent notes that there are limits to the types of diagnostic information available when

computer systems fail. To address such limitations, the invention in the ’683 patent “provides a method and apparatus to build a representation of the state of a computer, based on diagnostic data, by extracting system information from that diagnostic data and building a component based representation of the computer using the extracted system information.” Id. at 2:22-27. According to the ’683 patent abstract:

> A representation of the state of a computer, based on diagnostic data of the computer, is built by extracting system information from the diagnostic data and building a component based representation of the computer using the extracted system information. A static tree definition of a computer system is provided which is formed by element types in a fixed hierarchical relationship. A plurality of token types are provided, each of the token types being associated with one of the element types. The token types are component based data types. Respective segments of the incoming data that are defined by respective token types are identified and stored as tokens in a token data base. Each of the tokens has a value field holding a value associated with the element and a parent field referring to an element with which the token is associated. For each element in the static definition, the token data base is searched for associated tokens and a host state is built based on the static state definition and the extracted associated tokens, the elements of the static state definition being given value by their associated tokens.

The ’249 patent incorporates by reference the application for the ’683 patent. The ’249 specification repeats substantial parts of the specification of the ’683 patent and describes the same monitoring system claimed in the ’683 patent. It further describes a way “to generate alerts indicating predetermined conditions exist in a computer system.” ’249 patent at 2:28-31. That abstract states:

> A monitoring system generates alerts indicating predefined conditions exist in a computer system. Alerts are generated by comparing alert definitions to a host state representing the state of the hardware and software components of a computer system to determine if conditions defined in the alert definitions exist in the host state; and generating alerts accordingly. The host state is a static tree structure including elements in a fixed hierarchical relationship, the elements being given value by associated tokens, the elements and associated tokens representing the hardware and software components of the computer system. The alert definitions generate alerts according to the values of at least one token, at least one alert or a combination of various tokens and/or alerts. The host state is created by providing a static tree structure representing a general computer system. Component information indicating hardware and software components of the computer system is extracted from diagnostic data of the computer system. The host state is generated according to the static tree structure and the component information.

Claim 1 of the ’249 patent is a representative claim (disputed terms are in bold):

A method comprising:

> providing a host state representing a state of a **computer system**, the host state being represented as a **modifiable tree structure including elements in a fixed hierarchical relationship**, the elements being given value by associated **tokens**, the elements and associated **tokens** representing hardware and software components of the computer system and wherein the tokens are extracted from diagnostic data from the computer system;
>
> determining if predetermined conditions exist in the computer system by comparing respective definitions of the predetermined conditions to the host state; and
>
> generating an alert if one of the predetermined conditions is determined to exist.

’249 patent at 39:1-17.

1. “Token”

Disputed Claim Term: “Token” (’683 patent, claims 3, 4, 6, 7, 8, 10, 13, 16, 21, 22, 23, and 24; ’249 patent, claims 1, 8, 11,14, 16, and 18)	
NetApp’s construction	**Sun’s construction**
A data structure consisting of a name, an identifier of an element, an identifier of a test to be performed on the element, and a test output value.	Data that communicates information about a particular element.

The parties dispute whether a token consists of a four-part data structure. NetApp contends that it does, and Sun proposes a much more general definition of the term. To begin its analysis, the Court first turns to the claims themselves. The term “token” appears in claims 3, 4, 6, 7, 8, 10, 13, 16, 21, 22, 23, and 24 of the ’683 patent, and in claims 1, 8, 11, 14, 16, and 18 of the ’249 patent. Claims 1, 11, and 16 of the ’249 patent state that the host state is a tree structure including elements in a hierarchical relationship, with “the elements being given value by associated tokens, the elements and associated tokens representing hardware and software components of the computer system.” ’249 patent at 39:5-10, 40:29-33, 40:65-41:3; see also ’249 patent abstract. This claim language demonstrates that tokens give value to elements and represent hardware and software components of the computer system. Other claims also reflect that tokens communicate information about elements, but these claims vary in their specificity. See, e.g., ’683 patent at claim 6

("searching the token data base for at least one token that refers to the respective element type"); claim 16 ("at least one token respectively associated with each of the elements"); claims 3 and 21 ("each of the token types having a value of one aspect of the component information and an indication of an association with one of the elements in the static tree"); claim 23 ("at least one token that refers to the respective element type"). While this claim language indicates that tokens communicate information about elements, the claims do not specifically define what a token is. Nor do the claims rule out the possibility that a token may have other attributes, as evidenced by the varying specificity in which different claims discuss the relationship between tokens and the objects to which they refer. For example, claims 3 and 21 discuss tokens having a value of the component information and an association with an element, while other claims merely discuss the reference to an element. The claim language, therefore, neither defines tokens, nor precludes tokens having certain structural components.

Certain claims refer to the constituent parts of a "token." Claim 16 of the '683 patent, for example, refers to the test output value and the identifier of the element. However, as NetApp notes, reference to the constituent parts of a "token" in certain claims does not provide a basis for disregarding a definition provided by the specification. In Honeywell Int'l, Inc. v. Universal Avionics Sys. Corp., 488 F.3d 982, 990 (Fed. Cir. 2007), for example, the Federal Circuit upheld the district court's construction based on the specification, where the specification made clear that the disputed term "look ahead distance" was a function of both speed and time, even though the claim language described a signaling device for "defining a look ahead distance as a *function of the speed* of the aircraft" only. Id. (emphasis added). In the present case, therefore, even if certain claims only refer to certain parts of the token or describe tokens very generally, the token may still have a four-part data structure if the specification defines "token" to have such a structure. Accordingly, the Court turns to the specification.

NetApp relies heavily on the following definition of "token" in the specification in support of its proposed construction, while Sun disputes that the patentee was acting as a lexicographer in this passage:

> In order to extract information from the diagnostic data stream, "token types" are utilized. A token type **defines each token to have a token name and a test name**. A test name comes from the tests shown e.g., in Table 1 or in Table 2, and indicates which test output contains the information for each token. **In addition to a token name and a test name, each token has a label and a value**. The label for the token gives the token knowledge about what element the token is associated with, i.e., the parent of the token which is an element. The value of the token provides a value extracted from the diagnostic data that gives value to the element.

'249 patent at 7:10-20; '683 patent at 6:33-43 (emphasis added). NetApp argues that Sun's construction ignores this four-part structure of a token that includes: a name, an identifier of an element (token label), an identifier of a test to be performed on the element (test name), and a test output value (value of the token). In addition, NetApp argues that the following paragraph, which also appears in both patents, provides further confirmation that "token" refers to a four-part data structure:

> For instance, assume a disk element exists with a name of "c0t10d0." Assume also that a token exists for such a disk element indicating the number of sectors per cylinder**. The name of such a token** would be, e.g., "number of sectors per cylinder.**"** **The test name in the token** would be "vtsprobe" since the output of that test provides the information needed for the number of sectors per cylinder. **The label for the token** would be "c0t10d0" indicating that token is associated with a particular disk having that name. Finally, **the token would have a value** which indicates the number of sectors per cylinder.

'249 patent at 7:21-35; '683 patent at 6:44-58 (emphasis added).

The issue here is whether or not this four-part data structure limitation applies to all of the references to "tokens" in the patent claims. Statements that "describe the invention as a whole, rather than statements that describe only preferred embodiments, are more likely to support a limiting definition of a claim term." C.R. Bard, Inc. v. U.S. Surgical Corp., 388 F.3d 858, 864 (Fed. Cir. 2004) ("Statements that describe the invention as a whole are more likely to be found in certain sections of the specification, such as the Summary of the Invention."). Here, the passage describing the four-part structure is not in the summary of the invention. However, while statements that describe the invention as a whole are more likely to be found in certain sections of the specification of the invention, such as the summary of the invention (see id.), they need not be contained in only certain parts of the specification. For example, as NetApp notes, in Sinorgchem Co. v.

International Trade Commission, 511 F.3d 1132, 1136 (Fed. Cir. 2007), the Federal Circuit applied the definition of the term "controlled amount" that appeared in the "detailed description of the invention" in the middle of a long paragraph describing embodiments. See Weber Decl, Ex. 1 (U.S. Patent No. 5, 117,063 at 4:48-52). In Sinorgchem, the Court noted that the presence of quotation marks around the claim term strongly indicated that what followed was a definition. See 511 F.3d at 1136.

In the present case, the above passage defines the parts of a token. While the phrase "token" does not appear in quotation marks in this section of the specification, the phrase "token types" immediately proceeding the token term is in quotes, and the passage defines "tokens" in the context of "token types." In addition, the passage explicitly uses the word "defines" when outlining the token's components. Sun argues that the use of "define" has a specific meaning in computer science that refers to the creation of an object or variable in a programming language. Declaration of Dr. Hugh Smith ¶ 38. The same term "define" is used in a computer language sense in another part of the specification. See '249 patent at 9:42-48 ("an element can have a token defined that is the mathematical result of other tokens"). The sentence following the description of the token name and test name, however, states that "[i]n addition to a token name and a test name, each token has a label and a value," which indicates that the term "define" is used according to its normal usage in the passage at issue. '249 patent at 7:15-16. See also Reply Declaration of Professor Darrell Long ¶ 5 (noting that "define" is not used as a programming term in the patent). Even if the term "define" is being used in the computer language sense, however, a token would still require a four-part data structure under Sun's expert Dr. Smith's own reasoning. See Smith Decl. ¶ 38 (claiming that if the specification defined a token to mean integer, it would simply mean that tokens would have integer values); Long Reply Decl ¶ 5 (noting that under Dr. Smith's analysis, "because the specification states that token type

defines a token to be a four-part data structure, tokens will have that four-part structure").[1]

Furthermore, while other parts of the specification point out that certain descriptions are "exemplary," the passage defining "token" cited by NetApp does not contain such language. For example, the specification points out the exemplary nature of static trees discussed in the preceding passage, but not of the four-part structure of tokens. See, e.g., '249 patent at 6:5-6, 6:54-57. Rather, the passage defining the token components is itself followed by a more specific example, starting with "for instance," of what these specific components of a token may be. Id. at 7:10-36. This language further supports NetApp's assertion that its proposed definition of token is not limited to the preferred embodiment. Rather, the part of the specification on which it is based provides a high-level description of "token."

The specification then notes that there are two types of tokens: element realizing tokens, which provide a way to determine whether an element should be included when building a particular host state, and data tokens, which provide additional information about an element. This subsequent high-level description of tokens discusses the two types of tokens and reinforces the generalized nature of the "token" discussion. The specification then notes another "exemplary" output of a diagnostic test, before stating that the preferred implementation of the invention described is in an object-oriented computer language. Id. at 7:66-67, 8:8-10. Finally, the patent states that in "preferred embodiment the tokens in token data base 207 are stored as a hashtable to provide faster access to subsequent processing steps of building the representation of the system." '249 patent at 8:32-36. The above passages demonstrate that the patentee knew how to describe certain aspects of the invention as "exemplary" or "preferred implementations," yet did not so limit the four-part structure of tokens.

Sun relies on Tivo, Inc. v. Echostar Communications Corp., 516 F.3d 1290 (Fed. Cir. 2008), arguing that NetApp seeks to improperly limit the claim term. In that case,

---

[1] The Court recognizes the limited role of expert testimony in construing claims, but relies on it for background on the technology at issue and to analyze how a person of ordinary skill in the art understands various aspects of the patent. See Phillips, 415 F.3d at 1318.

Echostar argued that a construction requiring the use of object-oriented software was proper, because the patent described an embodiment that used terms characteristic of object-oriented programming. Id. at 1307. Tivo is distinguishable, however, because the applicant had not expressly defined a claim term. The Federal Circuit held that the use of an example that employs object-oriented programming is insufficient to require that the claims be limited to embodiments using such programming. Id. In contrast, the specification here defines the term "token type," and in that context, tokens themselves, in a higher level discussion of the term.

In addition, even if the specification describing the four-part structure did not provide an explicit definition, if the patentee uses the term throughout the entire patent specification in a manner consistent with this meaning, the term is defined by implication. See Bell Atl. Network Servs. v. Covad Commc'ns Group, Inc., 262 F.3d 1258, 1271 (Fed. Cir. 2001). Therefore, while the Court finds the lexicography relatively clear here, it turns to the other examples of tokens in the patent specification to examine whether they utilize a four-part data structure. The Court concludes that they do.

Turning to the various examples in the patent, as discussed above, Column 7 of the '249 patent discusses element realizing tokens. Sun argues that such tokens do not have a four-part data structure. Element realizing tokens "provide a way to determine whether an element should be included when building a particular host state." '249 patent at 7:42-50. A disk name token is a type of element realizing token that provides the parent field or the name. Element realizing tokens obtain information about the specific system to build a host state representation of the system. Hearing Tr. at 36. Such tokens, while not explicitly described as having a four-part data structure, follow the discussion of the four-part structure and fit into such a structure. As NetApp notes, the "test" for the disk name token would be the method by which the name is retrieved, and the name of the disk

constitutes the value.[2]

In addition, as noted above, the paragraph following the description of the four-part data structure gives an example of a data token that retrieves the number of sectors per cylinder, which explicitly has a four-part structure. Another subsequent example refers to at least three parts of that structure: "For example, another token associated with that disk element might be a disk manufacturer token that identifies the manufacturer as 'Seagate.' The value of the token in such an instance would be 'Seagate.'" '249 patent at 7:32-35. In this example, the token name would be disk manufacturer, and the label or element identifier would be the name of the disk, for example c01t10d0. While the patent does not explicitly discuss a test name for this example, the test name would be the name for the process used to retrieve the name, such as "get name," as Professor Long persuasively noted at the hearing. In other words, there is some type of operation or query by which the name "Seagate" must be retrieved. Because this example immediately follows the discussion of the four-part data structure, it utilizes a four-part structure, even though the patentee was emphasizing the value component, i.e., the manufacturer name, in this example.

The '249 patent discusses yet another example of a token, stating:

> An element can have a token defined that is the mathematical result of other tokens. For example, a disk space free token is derived from a simple subtraction from a disk used token and a total disk space token.

Id. at 9:42-49; '683 patent at 8:54-61. According to Sun, because the above mathematical operation is so basic, there is no "test to be performed" and no "test output value" generated. Smith Decl. ¶ 37. Thus, Sun contends that NetApp's construction improperly

---

[2] Sun maintains that NetApp's expert, Professor Darrell Long, provided examples of tokens that include only a two-part structure, giving the example of an element realizing token that identifies a disk drive and provides its name. Sun argues that a token that only provides information as to whether or not a disk exists in a certain location has only a one-part structure. However, as NetApp notes, the value component is really a value field for holding value information. This type of information will differ for different tokens. Therefore, even where the token is merely ascertaining whether or not a disk exists, the value field would still exist, but would not be populated in the ordinary sense of the word, as it would have a zero or blank value, as NetApp notes. In other words, regardless of whether or not the value is a name or a number or yes/no information, the value field exists to hold that type of information.

excludes this particular embodiment. While Dr. Smith argues that the basic mathematical operation to be performed for those tokens defined as a subtraction of the values of two other tokens does not generate a "test output value," the specification does not support his conclusion. Even a straightforward computer operation involving subtraction generates an output and involves a subtraction "test" or method, albeit a simple one. In fact, NetApp's expert, Professor Long notes that a person of ordinary skill in the art would understand that for a disk space free token, the test output value is the amount of free disk space determined through the subtraction, and the test to be performed is the subtraction operation. Long Reply Decl. ¶ 8. This is both logical and persuasive.

Another example of a token implementation involves tokens that are stored in a hashtable, which is a particular type of lookup table:

> In a preferred embodiment the tokens in token data base 207 are stored as a hashtable to provide faster access to subsequent processing steps of building the representation of the system. A hashtable is a common key/element pair storage mechanism. Thus, for the token hashtable, the key to access a location in a hashtable is the token name and the element of the key/element pair would be the token value. . . . Token types are run against the test output indicated in the test name in the token. For example token types having a test name parameter of "df" are run against "df" test output.

'249 patent at 8:28-47; '683 patent at 7:47-65. According to Sun, this portion of the specification does not indicate that an "identifier of a test to be performed" is stored in the hashtable. Sun argues that based upon the invention's design, there would be no reason to store an identifier of a test to be performed in the token database or hashtable, because at this point in the process, the relevant data has already been collected and properly associated with the correct element in the tree structure. Smith Decl. ¶ 39.

Sun's arguments are not persuasive, however. The hashtable discussion expressly repeats that the test name is part of the token. In addition, Sun is not really arguing that the hashtable example is at odds with a four-part data structure. Rather, Sun seems to maintain that the hashtable example has no need for all of the parts of the token that are in NetApp's definition. For example, while Sun's expert agrees that varying information can be

associated with the hashtable, Sun notes that because a hashtable could contain only yes/no information, this type of hashtable would not conform to the four-part structure. However, as NetApp notes, yes/no information is itself a type of value. See footnote 2, above. In sum, Sun's argument regarding hashtables is at odds with the discussion of hashtable in the specification, which discusses all four parts of the data structure and provides an illustrative test name and test output. In addition, Sun's own expert agreed that the hashtable could have more than two elements. Hearing Tr. at 23.

Sun argues that other parts of the specification support its own proposed construction. However, these portions of the specification actually reveal that Sun's own proposed definition is too general. For example, Sun notes that the abstract of the '683 patent states that data segments that "are defined by respective token types are identified and stored as tokens in a token data base. Each of the tokens has a value field holding a value associated with the element and a parent field referring to an element with which the token is associated." See also '249 patent abstract (noting that tokens are associated with elements and give the elements value). At the hearing, Sun conceded that each token is associated with an element, which is to say that it has a parent field, and Sun's expert stated that the invention requires a correlation between the element (i.e., parent field), and the value (i.e., value field). The abstract, therefore, shows that, contrary to Sun's proposed construction, a token must have at least two components: a value field and a parent field that refer to an element.

Sun also claims that the preferred embodiment description supports its definition, in particular the statement that the system received incoming diagnostic data from a monitored computer system and subsequently the test data "is processed by token processing 211 to extract the information associated with hardware and software components in the monitored system." '249 patent at 5:36-41; '683 patent at 5:4-11. The patent then notes that: "An element has tokens associated with it. Thus, a partition element may have a disk percentage token, disk name token, and space available token associated with it." '249 patent at 5:61-64; '683 patent at 5:26-32. However, this latter excerpt

shows that "elements" may be associated with tokens, each token capturing different aspects of the element, and the passage merely lists token types – it does not define the term token itself.

Finally, Sun argues that the '249 patent emphasizes the flexible and varying nature of tokens. See Sun Opposition at 7 (quoting '249 patent at 7:55-59; '683 patent at 7:9-13) ("The exact nature of the tokens and the total number of tokens will depend upon the system that is being monitored . . . ."). Just because tokens may be flexible and variable, however, does not mean that tokens do not share a common structure. The passages quoted by Sun discuss examples of token types and the purpose of tokens, but these passages do not provide a definition of token. In addition, the '683 abstract refers to two different parts of a token data structure, which is inconsistent with Sun's proposed construction of unstructured data. And, as NetApp notes, even though the abstract refers to certain parts of a "token," this reference does not provide a basis for disregarding the definition provided by the specification. Cf. Honeywell, 488 F.3d at 990.

Turning to the prosecution history, NetApp contends that the applicant made a statement during prosecution of the '249 patent reinforcing the four-part definition of token. Specifically, in response to the non-final office action filed on March 8, 1999, the applicant stated: "For a description of elements and tokens, please see, for example, pages 9-13 of the specification." See Nathan Decl., Ex. 5 at NAC0015585. NetApp argues that this patent prosecution history is similar to that in Irdeto Access, Inc. v. Echostar Satellite Corp., 383 F.3d 1295 (Fed. Cir. 2004), in which the Federal Circuit upheld the construction of the term "group keys" based in part on the applicant's statement during prosecution that the term had no accepted meaning in the art, but that the meaning was described in the specification. Id. at 1300, 1302-1303. The prosecution history here, however, does not inform the analysis of this term, because the parties are not arguing about whether the construction of "token" has an accepted meaning in the art apart from the specification, as

the parties argued in Irdeto.[3] Irdeto merely indicates that the definition of "token" may be found in the specification, but the parties here both agree that the specification defines the term (although they disagree as to the claim term's ultimate meaning).

Finally, according to Professor Long, all tokens must have the same number of fields to be consistent with the defined computer science format so that the system will work. Hearing Tr. at 57-58. While Sun's expert noted that this argument presupposes a four-part data structure, Dr. Smith did not dispute the fact that a computer system with a defined format must use that format in order to work properly and avoid crashing. Accordingly, the intrinsic evidence, most significantly, as well as the extrinsic evidence, on balance supports NetApp's proposed construction.

In sum, the Court finds that the token examples in the specification utilize a four-part data structure. However, NetApp's proposed construction is not entirely accurate for a number of reasons. First, as noted above, the "test" associated with each token is not necessarily a "test" in the normal sense. In other words, in computer language, this process may be called a test, but this does not necessarily comport with common usage, as it would likely be understood by a jury, so defining a token in this way could lead to juror confusion. Rather, the test is really a method by which the value is generated or by which the value field is populated. NetApp itself articulated the test as "deriving information about the particular system for which you want to build a representation." Hearing Tr. at 33. In addition, as Sun notes, the specification makes no reference to a test "to be performed" on the "element," as NetApp proposes in its construction. '249 patent at 7:10-15. In fact, the above example involving the subtraction of one token value from another to determine free disk space does not involve a test performed *on an element*.

---

[3] In Irdeto, the Court was addressing rejections based on indefiniteness. In response to the examiner's finding certain terms indefinite, the applicant stated that those terms were clearly defined in the specification. During claim construction, however, the patentee tried to argue that the term had an ordinary meaning in the art. The Court rejected this argument, relying on the prosecution history. Id. at 1298, 1300, 1303. In contrast, here, the patentee cited to pages 9-13 of the specification describing tokens and elements in order to overcome the examiner's rejection of certain claims as being unpatentable in view of prior art. The patentee was arguing that the description of elements and tokens are different from prior art. See Nathan Decl, Ex. 5 at NAC0015585.

Second, the portion of the specification relied upon by NetApp does not discuss a "test output value." Rather, it refers to a token having a value, which "provides a value extracted from the diagnostic data that gives value to the element." '249 patent at 7:17-20. Finally, the examples above make clear that the value field need not be populated, as NetApp itself conceded that a value could be zero or blank, i.e., empty.

In sum, the Court proposes construing "tokens" as "a data structure consisting of a name; an identifier of an element with which the token is associated; an identifier of the method by which the value field may be populated; and the value field, which holds the value associated with the element, which can include an empty value." While this is a proposed construction, the parties may only comment if they find a mistake or ambiguity in the wording, as opposed to disagreeing with the Court's reasoning, and they must do so within ten days of the date of this Order.

2. "Computer System"

Disputed Claim Term: "Computer System" ('683 patent, claims 1, 2, 6, 11, 12, 16, 18, 19, 20, 23; '249 patent, claims 1,2, 5, 7, 9, 10, 11, 12, 13, 14, 15, 16, 19)	
NetApp's construction	**Sun's construction**
A system containing one or more computers coupled in a network.	A system that includes at least one computer and that may contain a number of computers coupled in a network.

The parties agree that this term should receive the same construction in all of the claims of the '683 and '249 patents. See also Innova/Pure Water, Inc. v. Safari Water Filtration Sys., 381 F.3d 1111, 1119 (Fed. Cir. 2004) ("Unless otherwise compelled, when different claims of a patent use the same language, we give that language the same effect in each claim."). The parties also agree that a "computer system" contains one or more computers. The parties disagree, however, on whether that system must always be networked. The sole dispute involves the instance where the computer system includes just one computer, and whether in that instance, that computer must be "coupled in a network." Sun concedes that a computer system containing multiple computers is coupled in a

network, but asserts that a system containing one computer need not be networked. In this invention, there are two different aspects of networking. The monitored computer system may be coupled to other monitored computers in that same system via internal connections. In addition, the monitored system may be coupled to another computer system, e.g., the monitoring computer system. The latter type of connection is at issue here. Sun maintains that a single computer need not be connected to a monitoring system, while NetApp contends that it must.

As an initial observation, the patent uses the phrase "computer system" rather than the term "computer." As NetApp notes, the specification uses the terms computer system and computer differently. See, e.g., '249 patent at 3:66-4:2 ("The monitored [computer] system includes at least one computer and typically includes a plurality of computers."). NetApp argues that Sun's proposal erroneously construes computer *system*, because Sun's construction defines a single-computer system as a "computer," ignoring the "system" language in the claim term. NetApp's point is well taken.

NetApp's main argument is that the claimed invention of the '683 and '249 patents only makes sense if the computer system is connected or networked to other computer systems. NetApp Op. Brief at 7:15-17. Turning to the claim language, NetApp is correct that the majority of the claims require some type of communicating or monitoring between computers. Independent claim 1 of the '683 patent requires a first computer system that communicates diagnostic data to a second computer system. The next independent claim, claim 16, requires that the computer system be "monitored." Independent claim 19 refers to a monitoring computer system. Independent claim 25 requires a "monitored computer system" and a "monitoring computer system."

The independent claims of the '249 patent, however, do not all contain this communication language. See, e.g., claims 1-4 and 6-8 of the '249 patent. Sun's expert notes that claim 7, for example, uses the term "general computer system." Dr. Smith argues that the fact that the monitored computer contains a network is irrelevant, as the claim addresses what is contained in a modifiable static tree structure representing a

general computer system. Smith Decl. ¶ 48. However, Dr. Smith does not explain how the claimed method would be able to represent the computer system without communicating. In addition, all of these claims are method claims, which are not meant to spell out the system structure.[4] Furthermore, the ’249 patent incorporates by reference the ’683 patent, which refers to monitoring in each independent claim, and independent claims 10 and 16 of the ’249 patent require a “monitoring computer system.”

In addition, while a computer system generally is able to communicate, monitor, or be monitored by another computer system, the claim language itself does not specifically state a networking or a connection requirement. Rather, NetApp’s expert asserts that a person of ordinary skill in the art would understand that the connection between these systems is necessary for one to monitor another. Long Decl. ¶¶ 32-35. This conclusion is logical and unrebutted, but since there is no explicit networking requirement in the claim language, the Court turns to the specification.

NetApp correctly maintains that the specification shows that the “computer system” of these patents must be connected to other computer systems. First, the titles of both patents refer to remote monitoring and remote rebuilding of computer states, which require a connection to the system being monitored. The claimed invention in these patents is about remote monitoring, and therefore the computer system being monitored must be capable of communicating in order to be monitored. The field of the invention notes that the invention “relates to monitoring of computer systems.” ’249 patent at 1:20-24. In addition, the background of the invention states that it is “commonplace today” for a system to be part of a network, and that it would be “advantageous to provide a remote monitoring diagnostic system.” ’249 patent at 1:57-60, 2:14-15; ’683 patent at 1:52-53, 2:9-10. The summary of the invention of the ’249 patent provides that “the present

---

[4] Similarly, the specification itself first outlines an embodiment describing the method for providing a host state representing a state of the computer system and comparing alerts and generating alerts. ’249 patent at 2:31-37. While this brief description in the summary of the invention does not mention a monitoring system, again this embodiment is a method claim. The subsequent description of another embodiment is for a *monitoring* computer system apparatus, which describes the hardware for carrying out the previously described method. Id. at 2:54-60.

invention provides a method, apparatus and computer program products to generate alerts indicating predetermined conditions exist in a computer system," id. at 2:28-30, and that a person of ordinary skill in the art would understand from this description that a connection is a required part of the computer system of the patents. Long Decl. ¶¶ 33, 35-38. According to NetApp's expert, such a system would not work without connections between systems. Id. ¶ 38. Sun does not refute this point, and Professor Long's conclusion that connections between systems are required is persuasive.

In addition, the only embodiment disclosed in the specification includes a network connecting the monitoring and monitored computer systems. See figures 1a and 1b of both patents. In figure 1b, for example, even if one single computer were substituted for the plurality of computers "coupled in a network," that single computer would still be "connected" to the monitoring system.

At the hearing, NetApp also argued that an inventor, Mr. Chu, conceded at his deposition that the invention needs two systems. The inventor was discussing the term "remote" in the title of the patent and what he understood it meant. He was not interpreting the "computer system" phrase at issue as one skilled in the art would understand it. Sun notes that Mr. Chu did not specify whether he was referring to the claimed invention generally or to a preferred embodiment. While the Court may consider this inventor testimony only insofar as it relates to how one skilled in the art would understand the term "remote" in the context of remote monitoring, this testimony lends some further support to NetApp's assertion that even a single computer must be connected in such a way to allow remote monitoring. See Howmedica Osteonics Corp. v. Wright Med. Techn., Inc., 540 F.3d 1337, 1346-47 n.5 (Fed. Cir. 2008) (inventor testimony may be pertinent as to understanding the established meaning of particular terms in the relevant art, but inventor testimony concerning the scope of the claims is irrelevant to construction).

Sun also contends that its proposed construction is supported by the specification. However, Sun fails to distinguish internal versus external types of networking in its analysis of the specification. Specifically, Sun maintains that the patents provide near-

verbatim recitation of its construction: "The monitored system includes at least one computer and typically includes a plurality of computers 104, 106, 108, 110, and 112 coupled in a network as shown in FIG. 1b. . . . In exemplary computer system 100, which includes one or more computers and associated storage areas, preferably coupled in a network, incoming diagnostic data from monitored system 102 is received from modem 114 . . . ." '249 patent at 4:1-4:4, 4:9-13; '683 patent at 3:44-46, 3:51-54. Sun argues that the patents do not define computer system as including a single computer coupled in a network, and in fact expressly state that a network connection is only a preference. However, the above discussion of computers "coupled in a network" relates to the connections between computers in a monitored system, as opposed to the connection between a monitored computer system and the monitoring system. The specification makes clear that regardless of whether or not the single computer system is coupled in a network, that single computer must be able to receive incoming data "via email" or "direct modem connection" or "other communication channels." '249 patent at 4:10-17. In sum, the passages of the specification relied on by Sun merely state that the invention does not require a network between multiple computers in a monitored system, but it is the external type of connection between the monitored and monitoring systems that is in dispute here.

NetApp notes that this connection should be described as being "coupled in a network," as both parties use this phrase in their constructions and it is the term used in the art to refer to connections between computers. Sun does not argue that another term like "connected" should be used in lieu of "coupled in a network," should the Court agree that the disputed term requires connections between systems. Whenever computers communicate, they are connected by a network. Long Decl. ¶¶ 33-35; <u>see also</u> Microsoft Press Computer Dictionary at 327 (3d ed. 1997) (defining network as a "group of computer and associated devices that are connected by communications facilities") (Nathan Decl., Ex. 9 at NAC0109648). However, in light of potential jury confusion, and in light of the difference between connections between the monitoring and monitored system, the Court finds that using the phrase "coupled in a network" could mislead the jury. Rather, the

construction should make clear that where a computer system contains only one computer, that computer must be able to communicate via email or a dedicated modem connection or another communication channel to allow remote monitoring. See ’249 patent at 4:14-18.

In sum, NetApp is correct that the invention requires that even a single computer be networked or otherwise able to communicate with a monitoring system. Otherwise, the remote monitoring that is claimed would not be able to work. Sun does not tackle this argument head on. Rather, Sun argues that NetApp confuses the meaning of a “computer system” with the passing of information between two computer systems. Smith Decl. ¶ 49. Dr. Smith notes that NetApp appears to be confusing the term “computer system” with the need for the monitored computer system to communicate with the monitoring computer system. Smith Decl. ¶¶ 47-49. However, Sun does not grapple with NetApp’s point that the invention requires communication between the monitoring and monitored systems. In addition, both parties’ constructions address the connections of the computers in the computer system, and the Court finds it appropriate to address these connections in construing this term. Accordingly, the Court construes “computer system” as: “A system that includes at least one computer and that may contain a number of computers coupled in a network. Even where a computer system contains only one computer, that computer must be able to communicate via email or a modem connection or another communication channel to allow remote monitoring.”

///

///

///

///

///

///

///

///

///

3. "Modifiable Tree Structure Including Elements in A Fixed Hierarchical Relationship"/"Modifiable Static Tree Structure"

Disputed Claim Term: "Modifiable Tree Structure Including Elements in A Fixed Hierarchical Relationship"/"Modifiable Static Tree Structure" ('249 patent, claims 1 and 10/claim 7)	
NetApp's construction	**Sun's construction**
A tree structure comprising elements, which does not vary according to the system being monitored, and which can be edited by a user to add or delete elements representing hardware or software components on a computer system.	A tree structure that can be modified and that contains elements that are linked together in a hierarchical relationship, where the relationship does not change based on the systems being monitored. (Claims 1, 10) A tree structure that can be modified, but does not change based on the systems being monitored. (Claim 7)

The parties' dispute regarding this proposed claim term has been somewhat of a moving target. Originally, the parties' dispute centered around the terms "tree structure" and "modifiable." NetApp took issue with Sun's attempt to replace the term "tree structure" with the broader term "hierarchy." Because "tree structure" is another disputed term not ripe for construction at this point in time, Sun amended its proposed constructions to include the words "tree structure" (as does NetApp) in its construction instead of embedding a definition of that term in the construction. The sole remaining dispute concerns the term "modifiable" and whether that term requires that modifications be made by a user, as NetApp maintains, or whether the modifications also may be implemented through other means, as Sun maintains.

NetApp also contends that Sun's proposed construction, which states that the tree structure "can be modified but does not change," appears contradictory and will confuse the jury. At the hearing, Sun noted that NetApp's construction would be acceptable so long as the Court struck the phrase "edited by a user" and replaced it with "changed" or "can be changed." The Court agrees with NetApp that Sun's proposed construction is likely to confuse a jury, because the claim language "can be modified but does not change" is unclear as to how a structure can be both modifiable and static at the same time. The

Court therefore focuses on whether or not to adopt NetApp's proposed construction with or without the user-editing requirement.

As noted above, Claim 1 of the '249 patent provides:

> A method comprising:
> providing a host state representing a state of a computer system, the host state being represented as a **modifiable tree structure including elements in a fixed hierarchical relationship**, the elements being given value by associated **tokens**, the elements and associated tokens representing hardware and software components of the computer system and wherein the tokens are extracted from diagnostic data from the computer system;
> determining if predetermined conditions exist in the computer system by comparing respective definitions of the predetermined conditions to the host state; and
> generating an alert if one of the predetermined conditions is determined to exist.

'249 patent at 39:1-17. The reference to a modifiable tree structure also appears in claims 7 and 10 of the '249 patent. These claims make no reference to a "user," and the term "modifiable" seems to be used in a general sense, which supports Sun's construction, but the claim language does not define the term, and is not particularly instructive here.

Turning to the specification, Sun relies on the following excerpt to show that the specification discloses that a tree structure may be modified to add or remove elements corresponding to hardware and software components in the computer system being monitored, but does not place any restriction on who or what makes the modification:

> The description of the static tree is exemplary. Another tree may be chosen according to the system being monitored. **Additionally, the static tree may be modified to reflect hardware and software enhancements to computer systems.** The hierarchy tree definition is static in that it does not vary according to the system being monitored. However, the **hierarchy tree can be edited in element hierarchy editor 215 to accommodate additions and/or deletions from the hierarchy tree when for instance, a new technology begins to be utilized in the monitored computer systems**.

'249 patent at 6:54-63 (emphasis added). Sun is correct that there is nothing in the specification or claims requiring that such modifications be performed only by a user. This portion of the specification makes clear that the "element hierarchy editor" is simply an

embodiment, because it is described as "exemplary."[5] In addition, this "editor" requirement is not in the abstract, or the summary of the invention, and is not described as a necessary part of the invention. Therefore, the Court will not read this requirement into the claim term. Cf. Liebel-Flarsheim Co. v. Medrad, Inc., 358 F.3d 898, 906 (Fed. Cir. 2004) ("Even when the specification describes only a single embodiment, the claims of the patent will not be read restrictively unless the patentee has demonstrated a clear intention to limit the claim scope using 'words or expressions of manifest exclusion or restriction.'") (internal citation omitted).

Turning to the extrinsic evidence, Sun's expert maintains that it is just as likely that the supplier of the monitoring system would provide modifications to the static tree structure through software or firmware updates, rather than only through user edits. Smith Decl. ¶ 57. If a new hardware device is introduced into the marketplace – a CD-ROM, for example – and the static tree needs to be modified to allow a representation of that new device to be a part of it, the system could conceivably be programmed in such a way to allow such modifications through automatic updates.

NetApp's expert argues that "modifiable" must mean something more than automatic updates, because all systems have automatic updates. Dr. Smith contested this point at the hearing, noting that certain computer software components need to be sent to a vendor for modification, for example. Professor Long countered that such unmodifiable "software" is actually called firmware, which is no longer classified as software and is not implicated in the patent at issue. Hearing Tr. at 90. The experts' arguments are tangential. The patent makes clear that the static tree, while static, can be modified to reflect hardware and software enhancements. The use of the word "modifiable" in the claims focuses on the effect of the modification, rather than the process of the modification. Because the term

[5] Both parties' experts conceded that an "element hierarchy editor" requires that a user make those edits. Therefore, as to this particular embodiment, a human user is required. At the hearing, NetApp noted that the inventor Mr. Chu stated during his deposition that the editor needs to involve a human user. However, the inventor was asked about the element hierarchy editor, which Sun concedes requires a human user. This inventor testimony is irrelevant on the point of whether the term "modifiable" always requires a human user.

"modifiable" is not used to describe the process, NetApp's argument that the term "modifiable" is redundant is not persuasive.

In sum, the Court adopts a hybrid of the parties' constructions of the term "modifiable tree structure including elements in a fixed hierarchical relationship/modifiable static tree " and construes the terms as: "A tree structure comprising elements, which does not vary according to the system being monitored, and which can be changed to add or delete elements representing hardware or software components on a computer system."

B. '200 Patent

The '200 patent relates to network management and is titled "distinguished name scoping system for event filtering" and describes a "[m]ethod and system for allowing a computer network operations manager to subscribe for and receive notifications concerning network events from one or more objects or object levels, as defined by distinguished name scoping, and optionally having at least one event characteristic from a selected list." '200 Patent abstract.

The parties originally disputed the meaning of the phrase: "To examine [examining] the event communication to determine whether the event is associated with at least one object or object level on a selected object list having at least one specified object or object level." The parties, however, have now stipulated to a construction. The Court, therefore, adopts the parties' stipulated proposed construction, and construes the phrase as "to evaluate [evaluating] the event information received from a network device to determine if the event information relates to any item on a selected list of one or more network devices or network device levels."

C. '951 Patent

The '951 patent is titled "device centric discovery and configuration for fabric devices." Fabric devices are contained in fabric networks, which are a type of storage area network (SAN) that is capable of connecting a large number of network devices together. '951 patent at 1:57-60. The networks may have multiple paths between any two devices.

Id. Given the complexity of fabric networks, the patent describes a system for a host computer to discover and configure devices on a fabric network. Id. at 1:8-10. Specifically:

> A host may be coupled to a fabric network. Fabric devices attached to the fabric network may be visible to the host through one or more host adapter ports. The host system may include a device centric discovery interface configured to provide an interface to a fabric driver to obtain information about the devices in the fabric network. The device centric discovery interface may be configured to return device centric discovery information such that a multi-path fabric device is presented as a single device with transport information provided for each path to the multi-path device. A device centric configuration interface may provide an interface to the fabric driver for device centric configuration of the devices in the fabric for use by the host such that a requested fabric device is configured for use by the host on multiple paths in the fabric network.

’951 patent abstract.

Claim 1 is a representative claim (disputed term is in bold):

> 1. A system, comprising:
> one or more host adapter ports for coupling the system to a fabric network, wherein one or more devices attached to the fabric network are visible to the system through the one or more host adapter port;
>
> one or more processors configured to execute:
>
> a fabric driver configured to interface the system to the fabric network through the host adapter ports; and
>
> a device centric **discovery interface** configured to provide an interface to the fabric driver to obtain information about the devices in the fabric network, wherein the device centric **discovery interface** is configured to return device centric discovery information such that a multi-path fabric device is presented as a single device with transport information provided for each path to the multi-path device.

Disputed Claim Term: “Discovery Interface” (’951 patent, claims 1, 3, 6, and 9)	
NetApp’s construction	**Sun’s construction**
A software module that receives information directly from a fabric driver and provides the information to an application.	Software for discovering information about devices on a fabric network and providing the information in a specified format.

The parties have three main disputes: whether the language proposed by Sun –

United States District Court
For the Northern District of California

"for discovering information about devices on a fabric network" – should be included in the term's construction; whether the term "discovery interface" requires a direct connection to a fabric driver as proposed by NetApp; and whether the term should refer to an "application" as proposed by NetApp or a "specified format" as proposed by Sun.[6]

Regarding the first issue, Sun argues that the patent specification consistently describes discovery interfaces as providing the ability to discover information about the devices on a fabric network. For example, the summary of the invention section notes that device centric and transport centric discovery interfaces[7] "may provide an interface to the fabric driver to obtain information about the devices in the fabric network" and are "configured to provide" such an interface. '951 patent at 2:5-7, 21-24. See also id. at 6:50-52 (device centric discovery interface "may allow a user . . . to discover information about fabric devices . . ."); 6:65-68 (interface to fabric driver may include transport centric discovery interface "to provide fabric device discovery information"). While the specification describes this function of a discovery interface, this language is duplicative of the language already present in the claims. For example, claim 1 provides for "a device centric discovery interface configured to provide an interface to the fabric driver to obtain information about the devices in the fabric network." Id. at 14:25-28. Claim 6 also describes the discovery interface as one that obtains "information about the devices in the fabric network." Id. at 15:12-14. Claims 3 and 9 are dependant on claim 1. Therefore, while Sun argues that NetApp's construction is flawed due to its silence on this point, Sun's proposed description is redundant of language already present in the claims.

As to whether the term requires a direct connection, the claim language does not indicate that such a connection is required. Claims 1 and 6 call for a "discovery interface configured to provide an interface to the fabric driver," which indicates that the fabric driver is the object to which the discovery interface is connected. While one may infer that

---

[6] The parties originally disputed whether the interface is "software" or a "software module." At the hearing, however, the parties agreed that the interface was "one or more software modules."

[7] These two types of discovery interfaces differ in the format of the information sent and received through the interfaces. NetApp Op. Brief at 19, Sun Opp. Brief at 20.

this is a direct connection, the claim language by no means requires such an inference.

Turning to the specification, NetApp argues that the only examples in the specification show interfaces that are connected directly to the fabric driver and receive information directly from the fabric driver. In Figure 5, interface 503 contains discovery interfaces 520 and 522, as well as interfaces 524 and 526. In that figure, interface 503 is connected to fabric driver 504, and the figure depicts these connections with two arrows, one going from interface 503 to the fabric driver, and one going from the fabric driver to the interface. There is also an additional arrow pointing to the interface from the fabric driver labeled "events." The figure also shows a connection between interface 503 and administrative application 502, depicted by two arrows pointing from interface to application and vice versa. The written description describes interface 503 as providing "an interface between the administration application 502 and the fabric driver 504." '951 patent at 6:37-39.

NetApp's expert Professor Long maintains that a person of ordinary skill in the art would understand from this statement that the discovery interface is connected directly to the fabric driver and receives information directly from the fabric driver. Long Decl. ¶ 89. However, Sun's expert explains that one of ordinary skill in the art would conclude that the arrows in the software architecture diagram in Figure 5 indicate a connection, but not necessarily a direct connection between the connected components. Declaration of Dr. Martin Kaliski ¶¶ 35-36. In other words, while the components are conceptually connected and there is some mechanism by which the two components may communicate, the components need not be directly connected. In fact, at the <u>Markman</u> hearing, Professor Long himself conceded that the arrows in Figure 5 show that information flows in both ways, but do not necessarily reveal a direct connection; rather, the arrows are ambiguous in this respect.

The Court concludes that the specification does not require a direct connection. The arrows in Figure 5 portray a general conceptual connection. In addition, only interface 503 – not the discovery interfaces contained in the subset of interfaces inside of it – is

shown as connected to the fabric driver in Figure 5. Moreover, the arrows between the fabric driver and the interface are labeled events, indicating that the arrows show *what* is being communicated rather than the physical means by which the communication is accomplished. Finally, the description of Figure 5 states that the discovery interface "may" provide or return information, revealing that this figure is just one embodiment of the invention. See, e.g., '951 patent at 6:46-65 ("discovery interface may return device centric discovery information", "discovery interface . . . may provide . . . discovery information"). See also Liebel-Flarsheim Co., 358 F.3d at 906 (noting that even if the specification describes a single embodiment, the claims of the patent are not to be read restrictively absent clear intention to limit claim scope).

Sun also points out that the specification provides an example of an interface that does not provide such a direct connection: "the host system may include a fabric driver configured to interface the system to the fabric network through the host adapter ports." '951 patent at 2:2-11. While this portion of the specification discusses interfacing more generally as opposed to the "discovery interface" at issue, it indicates that the term "interface" need not involve a direct connection, because in the example given, the fabric driver is interfaced to the system, but is connected indirectly through host adapter ports. Since NetApp argues that the plain meaning of the term "interface" implies a direct connection as discussed below, this portion of the specification significantly weakens NetApp's position.

Sun also relies on the fact that the specification uses the term interface in the context of a "user interface," but these portions of the specification do not discuss whether such an interface is direct. '951 patent at 9:42-44, 10:1-3, 12:27-31. Sun's expert Dr. Kaliski notes that a user interface provides an interface between a user and an application, but connects that user to the application indirectly via intermediate components, such as a keyboard or mouse. Kaliski Decl. ¶ 30. This argument, while marginally helpful to Sun, is somewhat attenuated since the specification is discussing a different type of interface and does not focus on whether such an interface is directly connecting two things.

In looking at the claim language and specification as a whole, Sun is correct that the interface is not specifically described or defined as requiring a direct connection. See Kaliski Decl. ¶ 25. The specification simply requires that information about devices on fabric networks be discoverable, and indirect discovery is not explicitly ruled out. Id. The patent, therefore, does not support NetApp's argument for importing the term "directly" into this claim phrase.

The parties also argue over the significance of a statement by the applicant made during patent prosecution:

> In the current Office Action dated November 12, 2004, the Examiner states "it is clear that in the species of Fig. 4, the Fabric Driver (504) communicates directly with the Administration Application (502) whereas in the species of Fig. 5, the Fabric Driver (504) communicates with the Administration Application (502) via a dedicated interface (503)." However, the Applicant's specification does not state that the Fabric Driver (504) shown in Fig. 4 can only communicate **directly** with the Administration Application (502). The phrases "communicates directly" and "dedicated interface" used by the Examiner are not found in the Applicant's specification.

Williamson Decl., Ex. A (February 17, 2005 petition at 2-3) (emphasis in original). This portion of the prosecution history is a side detour that is not particularly helpful, because it involves Figure 4, which does not show the discovery interface, not Figure 5. NetApp points out that the patent examiner determined that the systems shown in Figures 4 and 5 were patentably distinct species, and therefore mutually exclusive. Weber Decl., Ex. 2 (August 3, 2004 Office Action at 2). The Examiner then directed the applicant to chose one of the two species for prosecution. The Examiner explained in the next office action that he made this determination because Figure 4 showed a direct communication between Fabric Driver 504 and Administrative Application 502, while Figure 5 showed communication through Interface 503. Weber Decl., Ex. 3 (November 12, 2004 Office Action at 2).

NetApp argues that applying the Examiner's reasoning to Figure 5, the Fabric Driver 504 must be directly connected to Interface 503, which includes the discovery interfaces. NetApp's argument is not persuasive for a number of reasons. First, as noted

above, the Examiner was discussing Figure 4, not Figure 5. Second, as Sun noted at the hearing, while the Examiner at one point stated that the patentee had to choose between the inventions claimed in the two figures, the Examiner nonetheless ended up allowing all claims. The Examiner's statements comparing the two figures, therefore, are largely irrelevant. Third, Figure 5 has arrows labeled events between the interface and the fabric driver, distinguishing it from Figure 4. Finally, the Examiner simply did not focus on whether anything other than the interface could be contained in the communication channel in Figure 5. If anything, this prosecution history supports Sun's argument that there is no requirement of a direct connection, as the applicant pointed out that such a phrase is not included in the specification.

The parties also rely on a great deal of extrinsic evidence – namely dictionary definitions – in support of their arguments. NetApp argues that the patent does not provide a specialized meaning for the word "interface," so the term is subject to the rule that "[i]n general, words used in a claim are accorded their ordinary and customary meaning." Cat Tech LLC v. Tubemaster, Inc., 528 F.3d 871, 884 (Fed. Cir. 2008) (using standard dictionary definitions to construe term) (internal citations omitted).

While NetApp's argument hinges largely on the common meaning of "interface," the proffered definitions do not require NetApp's construction. NetApp relies on a number of dictionary and technical dictionary definitions generally defining "interface" as a boundary: Webster's New World College Dictionary 3rd ed. 1997, p. 704) ("a plane forming the common boundary between two parts of matter or space"); The American Heritage Dictionary (2nd College Edition 1991, p. 669) ("a surface forming a common boundary between adjacent regions"); Microsoft Encyclopedia of Networking (2nd ed. 2002, p. 608-09) ("a mechanism for communicating between two devices" that "specifies the nature of the boundary between two devices and determines the procedures and protocols that make it possible for the devices to exchange data"). NetApp's expert notes that a person of ordinary skill in the art would understand that "just as a boundary is the edge of an object, that an interface is directly connected to and directly communicates with

the object." Long Decl. ¶ 87. However, as Sun notes, none of the proffered dictionary definitions include the word "directly" or any similar requirement. In addition, Sun offers its own secondary dictionary definitions that do not discuss a single boundary or direct connection between two objects. See Webster's New World College Dictionary (3rd ed. 1997, p. 704) ("a point or means of interaction between two systems"); Microsoft Computer Dictionary (5th ed. 2002, p. 279) ("software that enables a program to work with the user (the user interface, which can be a command-line interface, menu-drive interface, or graphical user interface), with another program such as the operating system, or with the computer's hardware").

The above definitions are not particularly helpful, as there is significant variation between them. In addition, some of NetApp's dictionary definitions apply to physical interfaces, not software, as software is an intangible series of computer instructions – not a physical object with a "surface," nor one capable of forming a "plane." Kaliski Decl. ¶ 34. The parties agree that the discovery interface is a software interface. For this reason, Dr. Kaliski maintains that one of ordinary skill in the art would not understand software to include a boundary. Id. Professor Long disagrees with this conclusion, giving a few examples of software interfaces that provide a boundary and direct connection. Long Reply Decl. ¶ 22. But Professor Long's opinion is significantly undercut by his concession that there may be software between the discovery interface and the fabric driver in this invention. Specifically, Professor Long stated that there could be "some steps in between [the discovery interface and the fabric driver] provided by something else," although he noted that such steps or code provide no additional functionality or transformation of the data. Hearing Tr. at 120-21. As Dr. Kaliski noted, Professor Long's conclusion that an interface requires a logical direct connection does not flow from NetApp's proffered dictionary definitions, which involve a physical connection. In fact, Professor Long acknowledged that attributing physical characteristics to software does not make sense. Id. at 120 ("physical-proximity requirement of instructions . . . doesn't make any sense inside a computer's memory"). In sum, neither the intrinsic nor the extrinsic evidence supports

including a directness requirement.

Turning to whether or not the phrase should be construed as providing information to an "application" or providing information in a "specified format," NetApp notes that Figure 5 shows that interface 503 (containing discovery interfaces 520 and 522) provides information to administrative application 502, yet Sun's proposed construction omits stating that the discovery interface provides information to an application. Sun responds that the proposed "application" requirement violates the doctrine of claim differentiation, because independent claim 1 includes no reference to an "application," but dependent claim 3 includes the limitation "wherein the processor is further configured to execute an application configured to request the device centric discovery interface to provide a list of fabric devices attached to the fabric . . . ." '951 patent at 14:38-43. However, claim 3 gives a much more detailed description of "application" – in fact, as NetApp points out, the claim adds nineteen lines of additional requirements to those stated in claim 1. It does not merely add a generalized "application" requirement. Therefore, reading this requirement into the term's meaning would not render the dependent claim superfluous. See Sinorgchem, 511 F.3d at 1139-40 (characterizing doctrine of claim differentiation as the "presumption that each claim in a patent has a different scope").

The Court agrees with NetApp that Sun's requirement that the discovery interface provide information "in a specified format," without stating what that format is, is confusing. Sun argues that reference to the specific format is clear, because of the surrounding claim language. Sun notes that the term "discovery interface" is prefaced by either "device centric" or "transport centric" in the claims. '951 patent at 14:24, 27, 41, 44, 15:10-11, 13, 26-27. Because these two types of discovery interfaces differ in the format of device information, the claims require that the discovery interface provide the information in a specified format, depending on the type of discovery interface. However, since the claims already specify that format, see, .e.g., claims 13 and 21 referring to "device centric format," Sun's proposed language is redundant and potentially confusing.

In sum, the Court adopts a hybrid of the parties' constructions of the term

"discovery interface " and construes the term as: "One or more software modules that receive information from a fabric driver and provide the information to an application."

D. '305 Patent

The patent is titled "dynamic parity distribution technique." The invention involves redundant array of independent disks ("RAID") technology, in which multiple disks are aggregated into a single storage volume. The patent relates to storage of parity, which essentially is redundant information used to correct mistakes. Specifically, the patent describes:

> A dynamic parity distribution system and technique distributes parity across disks of an array. The dynamic parity distribution system includes a storage operating system that integrates a file system with a RAID system. In response to a request to store (write) data on the array, the file system determines which disks contain free blocks in a next allocated stripe of the array. There may be multiple blocks within the stripe that do not contain file system data (i.e., unallocated data blocks) and that could potentially store parity. One or more of those unallocated data blocks can be assigned to store parity, arbitrarily. According to the dynamic parity distribution technique, the file system determines which blocks hold parity each time there is a write request to the stripe. The technique alternately allows the RAID system to assign a block to contain parity when each stripe is written.

'305 patent abstract.

Claim 1 is a representative claim (disputed term appears in bold):

> A system adapted to distribute redundant information across disks of an array, the system comprising: a storage operating system configured to invoke storage operations executed by a storage system, the storage operating system further configured to manage storage of information, including the redundant information and data, on blocks of the disks in response to disk access operations, the storage operating system including a storage module adapted to compute the redundant information in response to a layout of the data in stripes across the disks, the storage operating system maintaining at least one unallocated block per stripe for use by the storage module to store the computed redundant information, wherein the at least one unallocated block used to store the redundant information is located in any disk and wherein the location of the at least one unallocated block used to store the redundant information is dynamically allocated **in a non-fixed pattern** by the storage module before each write request is completed for each stripe.

Disputed Claim Term: "In a non-fixed pattern" (claims 1, 23-24, 33, 35-36, 39, 42, 44-45, 51)	
Sun's construction	**NetApp's construction**
In such a way that the [data and][8] redundant information can be placed within the stripe in any order.	In locations within the stripe that are not known in advance.

The parties dispute whether the term "non-fixed pattern" has a physical or a temporal connotation. Sun's construction takes into account the order and location of the placement of the information, while NetApp's construction focuses on when the locations are known.

Representative claim 1 requires that "the at least one unallocated block used to store the redundant information is located in any disk." Claim 1 at 12:18-20. While NetApp argues that this language means that the location of redundant information (also referred to as parity, which is a subset of redundant information) is not known in advance, the plain meaning of "located in any disk" refers to location – more specifically, that the parity can be stored in any disk. This language does not support a temporally-oriented construction. In addition, the claim language separately contains a timing component, as the claims require that the parity be dynamically allocated in a non-fixed pattern before each write request is completed for each stripe. See, e.g., Claim 1 at 12:22-24. This claim language further supports Sun's construction, which refers to the physical location of the parity, rather than NetApp's, which refers to when the location is known.

Turning to the specification, Sun first notes that the abstract states that any available block in a stripe can be used to store parity, and that the location of parity is assigned arbitrarily within the stripe:

> In response to a request to store (write) data on the array, the file system determines which disks contain free blocks in a next allocated stripe of the array. **There may be multiple blocks within the stripe that do not contain file system data (i.e., unallocated data blocks) and that could potentially store parity. One or more of those unallocated data blocks can be assigned to store parity, arbitrarily.** According to the dynamic

[8] At the hearing, Sun agreed to omit the words "data and" from its proposed construction.

> parity distribution technique, the file system determines which blocks hold parity **each time there is a write request** to the stripe. The technique alternately allows the RAID system to assign a block to contain parity when each stripe is written.

'305 patent abstract (emphasis added). The "Summary of the Invention" repeats the same language. Id. at 3:61-4:3. The arbitrary selection of the parity location is recited again in the context of the present invention. Id. at 6:24-37 ("In accordance with the present invention, the dynamic parity distribution system and technique distributes parity across disks of the array . . . There may be multiple blocks within the stripe that do not contain file system data (i.e., unallocated data blocks) and that could potentially store parity . . . One or more of those unallocated data blocks can be assigned to store parity, arbitrarily.").

The above language, which clearly applies to the invention as a whole, supports both parties' constructions. The specification confirms that any unallocated data block can be used to store parity, and that the location of parity is assigned arbitrarily within the stripe. It also confirms that the file system waits to determine which unallocated block will hold the parity until there is a write request to the stripe.[9] However, as noted above, the latter timing component is already contained in the claim language, which states that the information is allocated before each write request is completed for each stripe. See, e.g., Claim 1 ("is dynamically allocated in a non-fixed pattern by the storage module before each write request is completed for each stripe."). In addition, this existing claim language is clearer than NetApp's proposed construction, which, in reciting "in advance," does not explain "in advance" of what.

In addition, in the context of one embodiment, the specification notes that any available block within a stripe is a suitable candidate for either data or parity: "In the illustrative embodiment, the file system maintains at least one unallocated block per stripe for use by the RAID system. During block allocation, the file system provides an

---

[9] As NetApp notes, Figure 2 also illustrates this timing component via an order of steps. Figure 2 is a "flowchart illustrating a sequence of steps for distributing parity among disks . . ." The figure demonstrates that the first step in assigning parity is to determine which disks contain free blocks, the system then reserves as many free blocks as are required for the parity, identified the reserved blocks, and then assigns the parity to the reserved blocks. See '305 patent at Figure 2. While this flowchart supports NetApp's construction, the order of steps is discussed elsewhere in the claim.

indication to the RAID system of the unallocated block(s) to be used to store parity information. **All unallocated blocks on the disks of the array are suitable candidates for file system data or parity**. Notably, the unallocated block(s) used to store parity may be located in any disk and the location(s) of the unallocated block(s) can change over time." Id. at 4:4-12 (emphasis added). The specification then states that the location of parity within a stripe is determined arbitrarily, can vary from stripe to stripe, and is not determined by the RAID configuration. Id. at 6:16-23. The specification repeats several times that any unallocated block within a stripe can be selected to hold parity:

> According to the inventive technique . . . During block allocation, the file system provides an indication to the RAID system of the unallocated block(s) to be used to contain parity information. **All unallocated blocks on the disks of the array are suitable candidates for file system data or parity.** Notably, the unallocated block(s) used to store parity may be located in any disk and the location(s) of the unallocated block(s) can change over time. Moreover, **all blocks in a RAID group are available for potential allocation, since parity is not held in fixed locations**. In practice, this means that all blocks, including those that were "hidden" in the parity disk are available to the file system 160 for allocation in volume block number space.

Id. at 8:37-49 (emphasis added). This portion of the specification specifically contains the "fixed" location language. It strongly indicates that fixed location implies a physical location, i.e., that the data or parity can be placed in any unallocated block in the stripe. This language supports Sun's construction and indicates that the term means something more than NetApp's construction, which fails to state that the redundant information could be placed anywhere, so long as it is in an unoccupied block.

While NetApp argues that the "fact that parity is permitted to be stored in any location within a stripe, does not mean that in every system, at all times, parity must be stored on all of the disks," NetApp Opp. at 5-6, Sun's construction does not include such a requirement. Rather, Sun proposes that the parity "**can** be placed within the stripe in any order," not that it **must** be. In so arguing, NetApp concedes that parity may be stored in any location within the stripe. Its construction, however, omits this limitation, because it allows the claims to read on systems in which parity cannot be placed on any unallocated disk in the stripe. Sun gives the example that if a system permitted parity to be stored on

disk 1 or 2, but not disks 3, 4, or 5, but did not know in advance if the parity would be stored on disk 1 or 2, then such a system would satisfy NetApp's proposed construction, but would vitiate the patent's central teaching that parity may be stored on any disk. While NetApp argues that such a rule could not exist in the context of this patent, because the patent claims themselves require that the parity be able to be placed on "any disk," this language is not contained in all of the patent claims. See, e.g., Claims 36 and 42.

NetApp also argues that the non-fixed pattern does not refer to data or redundant information. However, as Sun notes, the claims themselves specifically refer to the arrangement of redundant information and data. For example, claim 23 provides for "the system of claim 1, wherein the non-fixed pattern is created by the redundant information being stored in any block remaining after the data is allocated to blocks of the stripe." The "non-fixed" pattern refers to relative placement of data and redundant information within a stripe. Sun's original proposed construction, however, implied that the data blocks themselves are ordered arbitrarily like the redundant information, which is not supported by the patent. In fact, Sun itself concedes that its construction "does not require arbitrary data block assignments," but rather "simply refers to the relationship between redundant information and data blocks – without requiring a particular ordering of blocks within a stripe." Sun Reply at 4-5. According to the patent, the parity is distributed in a non-fixed pattern which depends upon placement of the preexisting data, as the parity is only placed in unallocated blocks. A more accurate construction, therefore, would refer to placement of redundant information "in such a way that the redundant information can be placed within the stripe on unallocated blocks in any order."

At the hearing, NetApp articulated its argument in a new way, stating that the pattern is non-fixed with regard to the system as a whole, i.e., the patterns across the stripes, but the pattern need not be arbitrary within a stripe.[10] NetApp argued, as an example, that the invention would encompass an algorithm which would store parity in

[10] NetApp also argued that the system is designed to balance the load (see, e.g., '305 patent at 9:41-45) and that a completely random placement of parity does not make sense in the context of this invention, but Sun's construction does not require a completely arbitrary assignment.

every third free block of the system. However, NetApp's expert Dr. Gregory Ganger conceded that when applied to an empty system, this algorithm would yield a system-wide non-random pattern. Only after time, according to Dr. Ganger, would this pattern become random, because the previously stored data itself would be changing and would exist in a random pattern. But the patent requires that parity be permitted to be stored in any location within a stripe, as NetApp concedes in its brief. Opp. at 5:27-28. The patent does not limit this requirement to a mature system. Yet in the above example of an immature system, under NetApp's proposal, parity would be located in a fixed pattern across the stripes, which the patent disavows.

As to the prosecution history, Sun argues that the applicants stated four separate times to the PTO that the invention permits the placement of parity and data in any order in a stripe. First, after the PTO rejected certain claims as being anticipated by the Patterson reference, NetApp contrasted Patterson by stating that the applicant's invention stores parity data in the unallocated blocks in a stripe, and that the "unallocated blocks can change with each stripe and do not follow a constant pattern." Williamson Decl., Ex. E (May 22, 2006 response to PTO) at 14. The PTO again rejected certain claims as being anticipated by Patterson, and NetApp responded by adding the "dynamically allocated" claim language and argued that "the storage operating system then selects at least one unallocated block of the stripe to store redundant information. This allows for the redundant information to be placed in any order, varying from stripe to stripe." Id., Ex. F (Oct. 23, 2006 response to PTO) at 13-14. In contrast, according to the applicant, Patterson described a RAID system where parity information is stored in a rotating pattern. The PTO again rejected claims as being anticipated, and NetApp responded on February 5, 2007. NetApp responded by adding the "in a non-fixed pattern" language to the claims and argued:

> In further detail, Applicant's invention allows **the at least one unallocated block used to store the redundant information is dynamically allocated by the storage module before each write request for each stripe in a non-fixed pattern.** Before each stripe is written across each disk, the storage module determines at least one unallocated block to use for redundant information. Each unallocated

> block is selected from the one or more blocks that are not already allocated to data blocks. The storage operating system then selects at least one unallocated block of the stripe to store redundant information. **This allows for the redundant information to be placed in any order, varying from stripe to stripe because there is no fixed pattern.**

Williamson Decl., Ex. G (Feb. 5, 2007 response to PTO) at 13-14 (emphasis in original). The PTO rejected the claims again as being anticipated, and NetApp moved the location of the "a non fixed-pattern" language in the claims and argued that the invention allows "the at least one unallocated block used to store the redundant information is dynamically allocated in a non-fixed pattern by the storage module before each write request is completed for each stripe." Williamson Decl., Ex. H (June 28, 2007 response to PTO) at 16-17.

The PTO issued its Notice of Allowance on September 7, 2007, and in his reasons for allowance, the Examiner stated: "The prior arts . . do not teach . . . a method, system, program and an apparatus that in which at least one block is dynamically assigned/allocated in a **non-fixed pattern (in any disk)** for storing the parity/redundant information/data **before** completing each write request for each stripe." Williamson Decl., Ex. I (Sept. 7, 2007 Notice of Allowance) at 5 (emphasis in original).

The Notice of Allowance, therefore, shows that the Examiner expressly identified as a point of novelty that the redundant information is stored in a non-fixed pattern that can be on any [unallocated] disk on which the stripe is written in any order. The Examiner also identified as a point of novelty that this dynamic assignment/allocation occurs before each write request for each stripe is completed. However, as discussed above, this latter timing requirement is already included in the claim language.

In response to this prosecution history, NetApp argues that the applicants' statements to the PTO must be read in context. But the prosecution history cited above includes the portions relied upon by NetApp and teaches that redundant information may be placed in any order on any disk. The prosecution history, therefore, supports Sun's construction.

In sum, Sun's construction, with minor adjustments, is more appropriate. NetApp's

construction only addresses the timing of when locations in the stripe are known, which is already addressed by other parts of the claim language, and fails to discuss the physical placement of the redundant information. For these reasons, the Court construes "non-fixed pattern" as "in such a way that the redundant information can be placed within the stripe on unallocated blocks in any order."

E. ’152 Patent

This patent relates to the use of groups and is titled "consistent logical naming of initiator groups." The invention describes:

> A technique enables efficient access to logical unit numbers (luns) or virtual disks (vdisks) stored on a storage system, such as a multi-protocol storage appliance. The technique allows a grouping of initiators by a "human-friendly" logical name that is mapped to a lun or vdisk on the storage appliance. The initiators are clients operating in, e.g., a storage area network (SAN) environment that initiate requests for the vdisk using block-based access protocols, such as the Small Computer Systems Interface (SCSI) protocol encapsulated over TCP/IP (iSCSI) or over fibre channel (FCP). The technique enables access to the vdisk by all initiators that are members of the initiator group (igroup). An igroup is a logical named entity that is assigned to one or more addresses associated with one or more initiators. These addresses may comprise fibre channel (FC) world wide name (WWN) or iSCSI name identifiers (IDs). Therefore, rather than having to specify these IDs when desiring access to a vdisk, an initiator need only specify the human-friendly name of the igroup.

’152 patent abstract. A main advantage of this invention, therefore, is that an "igroup" may be used to provide convenient access to a vdisk obviating the need for more cumbersome identifiers.

Claims 8 and 25 are representative claims and provide:

> Claim 8. A storage operating system configured to enable efficient access to a virtual disk (vdisk) stored on a storage device of a storage system, the storage operating system comprising: a user interface adapted to receive commands that create an initiator group (igroup) of initiators and map the vdisk to the igroup, the commands specifying a user selected igroup name, addresses of the initiators and a logical unit number (lun) identifier (ID) assigned to the vdisk; a file system configured to provide volume management capabilities for use in block-based access to the vdisk stored on the storage device; a vdisk module that cooperates with the file system to bind the user selected igroup name to the initiator addresses; and a small computer systems interface (SCSI) target module that cooperates with the vdisk module to map the user selected igroup name to the vdisk, the SCSI target module implementing a mapping function that allows the vdisk to be exported to the initiators of the igroup.

> Claim 25. A method for faster access to a virtual disk (vdisk) stored on a

> storage system, the method comprising: grouping one or more clients into an initiator group (igroup) entered through an interface, where the one or more clients initiate a request for the vdisk using a blockbased access protocol; selecting a selectable name for the igroup, where the igroup is a logical named entity that is assigned to one or more addresses associated with the one or more clients; and mapping the vdisk to the user selected name for the igroup to allow the one or more clients to access the vdisk using the selectable name.

Disputed Claim Term: "Initiator group (igroup)" (claims 2, 8, 18, 21, 25, 26, 29, 33)	
Sun's construction	**NetApp's construction**
A logical named entity with a human-friendly name that is assigned to one or more addresses associated with one or more initiators. Membership in the entity can be modified at any time by adding or removing initiators.	This term does not need to be construed because its plain meaning suffices and this term is already clearly defined in the body of the claims at issue. If the Court determines that this term requires construction, it should be construed to mean "a named set of one or more addresses of clients that can initiate requests."

The parties dispute whether or not the claim term needs construction in the first instance. Insofar as the Court agrees that the term needs construction, the parties' constructions present two main disputes: whether an igroup must have a human-friendly name, and whether an igroup must be capable of being modified by adding or removing initiators. Sun contends that igroups must have these characteristics.

As to the first issue, NetApp notes that the claims already define "igroup." Claim 25, cited above, describes "selecting a selectable name for the igroup, where the igroup is a logical named entity that is assigned to one or more addresses associated with the one or more clients." NetApp argues that this definition of igroup – "a logical named entity that is assigned to one or more addresses associated with the one or more clients" – is sufficient and complete because it explains that an igroup is assigned to one or more clients and the igroup is itself named, which permits the benefit described in the patent of referring to the igroup by its name, eliminating the need to refer to individual clients. However, as Sun points out, this definition is not included in all of the claims in which the term appears, see, e.g., Claim 8, which could mean that the term as it is used throughout the claims has a different meaning than the definition set forth in certain specific claims.

NetApp relies on TIP Systems, LLC v. Phillips & Brooks/ Gladwin, Inc., 529 F.3d 1364, 1369 (Fed. Cir. 2008), noting that the Federal Circuit upheld the District Court's construction where it relied "heavily on the claim language to construe the claim term." In that case, the court's construction was "supported by an identical definition in the specification." Id. Here, too, the specification states: "An igroup is a logical named entity that is assigned to one or more addresses associated with one or more initiators." '152 patent at 2:42-45. The use of the word "is" in the specification may "signify that a patentee is serving as its own lexicographer." Sinorgchem Co. v. ITC, 511 F.3d 1132, 1136 (Fed. Cir. 2007) (citation omitted).[11] TIP Systems is somewhat distinguishable, however, as appellant there was arguing for a construction that was contrary to the express language in the claim and specification, while Sun's proposed definition would supplement (with additional limitations), rather than contradict, the definition contained in the specification and the claim. See 529 F.3d at 1369. Despite this distinction, the Federal Circuit's analysis supports NetApp's proposed construction, but the Court turns to other portions of the specification for further guidance.

Sun argues that the summary of the invention requires the human-friendly limitation. See C.R. Bard, Inc. v. U.S. Surgical Corp., 388 F.3d 858, 864 (Fed. Cir. 2004) ("Statements that describe the invention as a whole, rather than statements that describe only preferred embodiments, are more likely to support a limiting definition of a claim term . . . . Statements that describe the invention as a whole are more likely to be found in certain sections of the specification, such as the Summary of the Invention."). The summary of the invention begins:

> The present invention overcomes the disadvantages of the prior art by providing a technique that enables efficient access to logical unit numbers (luns) or virtual disks (vdisks) stored on a storage system, such as a multi-protocol storage appliance. **The technique allows a grouping of initiators by a "human-friendly" logical name that is mapped to a lun or vdisk on the storage appliance.** By "human-friendly" it is meant, generally, a hierarchical naming convention that may use a spoken language name including an arbitrary label selected by a user or administrator. . . .

[11] Here, unlike in Sinorgchem, the patentee did not set off the term "igroup" in quotation marks, which are an indicator of an explicit definition.

> The inventive technique enables access to the vdisk by all initiators that are members of the initiator group (igroup). An igroup is a logical named entity that is assigned to one or more addresses associated with one or more initiators. These addresses may comprise fibre channel (FC) world wide name (WWN) or iSCSI name identifiers (IDs). **Therefore, rather than having to specify these IDs when desiring access to a vdisk, an initiator need only specify the human-friendly name of the igroup.**

’152 patent at 2:26-48 (emphasis added).

The summary then states: “According to the invention, the technique includes a method of creating logical igroups of initiators, **each identified by a human-friendly name or label**, and binding of each created igroup to one or more WWN or iSCSI IDs.” Id. at 2:49-52 (emphasis added). The abstract, excerpted in the above description of the ’152 patent, also discusses the human friendly name, noting that the technique of the invention allows a grouping of initiators by a human friendly logical name, so that an initiator need only specify the human friendly name of the igroup when desiring access to a vdisk.

The detailed description portion of the specification further states:

> The present invention relates to a technique that allows a grouping of initiators by a "human friendly" logical name that is mapped to a lun or vdisk stored on the multi-protocol storage appliance to thereby enable access to the vdisk by all initiators that are members of the initiator group (igroup). As used herein, a "human friendly" logical name is an arbitrary label selected by the user of administrator that may be a spoken name, a path designation or include a hierarchical naming convention. An exemplary human friendly name would be "administrators" for a name of an igroup that comprises the administrators of a given network.

Id. at 10:36-47. See also id. at 10:54-58 (rather than having to specify certain name identifiers when desiring access to a vdisk, “an initiator need only specify the human friendly name of the igroup”); id. at 12:17-21 (“According to the invention, the novel technique includes a method of creating logical igroups of initiators, each identified by a human friendly name or label”).

Sun argues that each of these statements in the specification establishes that the invention itself, not merely a preferred embodiment, consists of an igroup with a human friendly name. However, the specification also makes clear that the term “human friendly”

does not mean user-friendly as perhaps a juror would understand the term. Rather, it has a specific meaning, which is a "hierarchical naming convention that may use a spoken language name including an arbitrary label selected by a user or administrator" or "an arbitrary label selected by the user of [sic] administrator that may be a spoken name, a path designation or include a hierarchical naming convention." '152 patent at 2:32-35, 10:36-47. A path designation, for example, is often not a simple English, user-friendly name. Therefore, at a minimum, if the construction contains the "human friendly" limitation, it must also define that term so as not to confuse a jury.

In addition, as NetApp's expert points out, the specification discusses "allowing" grouping initiators by a human friendly name, suggesting that their use is not required. Ganger Decl. ¶ 34. However, as Sun notes, the phrase "allow" is not used in every instance cited above. In addition, despite using the word "allow" in some instances, the specification still seems to be describing the nature of the invention as a whole. "Human friendly," therefore, seems to be part of the definition of "igroup." The claim language and specification, when taken together, show that the claimed invention, at a minimum, permits a human friendly name for igroups.

NetApp also argues that the claim language and its history reveal that the "human friendly" limitation is not a *requirement* of the invention. The claims themselves never mention "human friendly," and instead only require that an igroup name be "selectable." See, e.g., '152 patent at 18:7 (Claim 25). Dr. Ganger notes that the claims used to contain the term "human friendly," but were amended to "user selected," which was then changed to "selectable." Ganger Decl. ¶ 34. In other words, "human friendly" was explicitly removed from the claims, which is strong evidence that the limitation should not be imported into the construction. When the applicant filed the application for the '152 patent, some of the original claims (e.g., independent claims 1 and 6, which became 8) included the term "human-friendly." See Weber Decl., Ex. 9 (U.S. Patent Application) at NAC0000940-44. Others (e.g., independent claims 16 and 19, which became claims 18 and 21, respectively) instead simply used the phrase "initiator group (igroup) name." Id.

To one of ordinary skill in the art, this history would have indicated that, for some claims, a human-friendly name was required for the initiator group and, for others, it was not. Further, in the April 4, 2006 amendment, the applicant changed several occurrences of "human-friendly" in the claims to "user selected," explicitly removing the human-friendly name requirement from the subset of claims that originally included it. Weber Decl., Ex. 10 (April 4, 2006 Amendment) at NAC0001656-63. The applicant later changed two instances of "user selected" to "selectable." Weber Decl., Ex. 11 (August 25, 2006 Amendment) at NAC0001683-91. Ganger Decl. ¶ 34. The patent prosecution history, therefore, indicates that the human friendly limitation may be a permissive limitation, but is not a requirement of the invention.

The Federal Circuit's analysis in Liebel-Flarsheim Co. v. Medrad, 358 F.3d 898 (Fed. Cir. 2004), sheds some light on the meaning of the specification and the prosecution history here. In Liebel-Flarsheim, the Federal Circuit considered the scope of claims in patents related to fluid injectors used during medical procedures. Id. at 900. The Court found that the language in the patent abstract did not suggest that a pressure jacket was an essential component of the invention, nor was there language in the specification that disclaimed the use of the invention in the absence of a pressure jacket. Id. at 908 (where abstract stated that an injector and method of replacing the injector was provided "in which the syringe is loadable and unloadable into and from the injector through the open front end of a pressure jacket of the injector," and the summary of invention stated that "according to the principles of the present invention, there is provided an . . . injector having a front end loadable syringe that can be loaded into and removed from the injector pressure jacket . . ."). The present case is in a somewhat different posture, insofar as the patent here distinguishes prior art on the basis that it required specifying various name identifiers when desiring access to a vdisk, unlike the igroup. See '152 patent at 2:26-48. Therefore, the specification in the present case provides somewhat stronger evidence for limiting the claim term than did the specification in Liebel-Flarsheim.

However, in Liebel-Flarsheim, the patent applicants specifically replaced claims

with references to a pressure jacket with a new set of claims, many of which omitted that limitation. Id.at 909. This prosecution history is quite similar to that of the '152 patent. The Federal Circuit noted that the replacement was a "**strong** indication that the applicants intended those claims to reach injectors that did not use pressure jackets." Id. (emphasis added). While the prosecution history in that case contained an explicit statement of intention (lacking in this case), as applicants stated that there was not necessarily a pressure jacket in the amended claims, id., in the present case, the omission of the "human friendly" term from the claims (and replacement of that term with "user selected" and then "selectable") provides similarly strong evidence that the applicant intended to broaden its claims.

After examining the patent prosecution history, the Liebel-Flarscheim Court noted that "[t]he only remaining question is whether the applicants failed in their effort and the pressure jacket limitation remained a part of all of the claims, even those from which the reference to the pressure jacket had been removed." Id. The Court, however, was unpersuaded by that argument, especially in light of the express statement in the prosecution history. Finally, the Federal Circuit also held that "[t]he fact that a patent asserts that an invention achieves several objectives does not require that each of the claims be construed as limited to structures that are capable of achieving all the objectives." Id. at 908. Similarly, here, there are multiple objectives of the invention, and each claim need not achieve all of them. Applying Liebel-Flarscheim's analysis to the present case, the Court finds that the prosecution history, when read with the specification and claim language, indicates that igroups may, but need not, have a "human friendly" name.

As to the second issue – whether an igroup must be capable of being modified by adding or removing members from the group – Sun argues that an igroup cannot be static and must be capable of modification. According to Sun, NetApp's construction would permit an igroup that is incapable of being modified once it is created.

The claim language does not require or mention this "modification" characteristic.

The parties' arguments primarily hinge upon the specification. NetApp notes that the summary of the invention contains a definition of igroup, which omits any reference to being modifiable. The summary states: "An igroup is a logical named entity that is assigned to on or more addresses associated with one or more initiators." '152 patent at 2:42-45. Sun argues that the summary of the invention also supports its proposed construction, citing the following portion of the summary:

> An igroup has certain attributes, such as transport protocol type and operating system type of the member initiators. Illustratively, the igroup need not be homogeneous in terms of these attributes, i.e., an igroup can contain initiators having different combinations of FCP and/or iSCSI as a transport. For example, iSCSI and FCP initiators can be combined into a single igroup. In addition, igroup can support various operating system initiator members. This allows operations, such as graceful rolling upgrade of a FCP SAN cluster to an iSCSI cluster, with no application downtime. Moreover, membership of the igroups can be modified at any time, i.e., initiators can be added to or removed from an igroup and, as a consequence, inherit or lose the mappings of the igroup, respectively.

'152 patent at 2-64-3:10. See also id. at 3:16-20 ("e.g., when replacing an initiator in a client"), 15:18-22 (same); 14:29-33 (discussing illustrative embodiment, noting "[i]n addition, membership of the igroups can be modified at any time, i.e., initiators can be added to or removed from an igroup.").

NetApp argues that the above paragraph makes clear that the listed features, including the modifiable feature, are examples, as the paragraph uses the terms "illustratively" and "for example." However, the word preceding the modifiable discussion is "moreover," which means in addition to what has been said. Thus, the paragraph is ambiguous as to whether or not the "modifiable" feature is a necessary aspect of the igroup, or is merely an example of a characteristic of an igroup. But it is clear that the modifiable feature is just one of the many listed attributes.

Sun notes that NetApp does not challenge the facts that: (1) there is no teaching in the patent of a static igroup, and (2) the teaching of the patent is to the contrary. At the hearing, however, the parties discussed whether or not the patent contained an example of an igroup that was not capable of being modified. NetApp noted that the address change example in the patent involves an igroup that is associated with multiple vdisks and is not

modified. When the initiator that is bound to the igroup is replaced, only the initiator address is changed. Therefore, the initiator address, as opposed to the igroup itself, is modified, which requires no additions, changes, or deletions to the igroup itself. See ’152 patent at 14:34-60. The address change example, therefore, weighs slightly in favor of NetApp’s construction, because it is an example in which the igroup does not change and the membership itself is not modified.

Sun also relies on extrinsic evidence, noting that the term “group” has a well-understood meaning in computer science consistent with its construction, as it is well-understood by those of ordinary skill in the art that groups can be modified by adding or removing members of the group. Declaration of Dr. Scott Brandt ¶ 49. Conversely, a single-member set that is incapable of adding or removing members would not be considered to be a group. Id. ¶ 50. Dr. Ganger, however, contests this point, noting that to one of ordinary skill in the art, a “group” is simply a named set of one or more things (its members). While groups usually allow modification of membership, this feature is not a definitional aspect. Also, neither the Microsoft computer dictionary definition nor the BSD book excerpt provided by Sun’s expert Dr. Brandt mentions that groups are modifiable. See Ganger Decl. ¶ 36; Brandt Decl. ¶¶ 48-49 & Ex. 3. Accordingly, the Court does not find Dr. Brandt’s opinion to be persuasive and does not find that an igroup must be capable of being modified.

In sum, the Court construes the term “initiator group (igroup)” as “a logical named entity that may have a human friendly name assigned to one or more addresses associated with one or more initiators. As used in the patent, ‘human friendly name’ means an arbitrary label selected by the user or administrator that may be a spoken name, a path designation or include a hierarchical naming convention.”

F. ’351 and ’097 Patents

The patents are each titled “enforcing uniform file-locking for diverse file-locking protocols.” The ’097 patent is a continuation of the ’351 patent. They share the same specification and are directed to the same file-locking technology. For convenience, the

Court largely cites to the specification of the ’351 patent only.

These patents address the issue that arises when more than one client accesses the same file on the file server at the same time. The technique relevant to the patent is locking, in which applications synchronize by acquiring locks from the file server. The invention is geared to allow file and data-sharing among clients that use diverse or incompatible file-locking protocols. Specifically:

> The invention provides a method and system for correct interoperation of multiple diverse file server or file -locking protocols, using a uniform multi-protocol lock management system. A file server determines, before allowing any client device to access data or to obtain a lock, whether that would be inconsistent with existing locks, regardless of originating client device or originating protocol for those existing locks. A first protocol enforces mandatory file-open and file-locking together with an opportunistic file-locking technique, while a second protocol lacks file-open semantics and provides only for advisory byte-range and file-locking.[12]

’351 patent abstract.

Claims 1 of the ’351 and ’097 patents are representative independent claims. Claim 1 of the ’351 patent provides (disputed terms in bold):

> A method of operating a file server, said method including steps for enforcing a **uniform file-locking semantics** among a set of client devices using a plurality of diverse file-locking protocols, said file server implementing said plurality of diverse file-locking protocols and enforcing said **uniform file-locking semantics** for said plurality of diverse file -locking protocols;
> wherein said **uniform file-locking semantics** includes steps for granting an **opportunistic lock** on a selected file to a first said client device in response to a first message using a first protocol;
> and breaking said **opportunistic lock** in response to a second message using a second protocol;
> wherein the first protocol and the second protocol are ones of said plurality of diverse file -ocking protocols.

Claim 1 of the ‘’097 patent provides:

> A method of enforcing **uniform locking semantics** among a set of client devices that use a plurality of diverse locking protocols, comprising the

[12] The patent sometimes refers to “file-locking” and sometimes refers to “file locking.” Since this discrepancy appears to be a distinction without a difference, the Court uses the hyphenated version of the term throughout this opinion.

steps of:
granting an **opportunistic lock** on a selected data set to a first client device in response to a first message using a first protocol; and
breaking said **opportunistic lock** in response to a second message using a second protocol;
wherein the first protocol and the second protocol are different ones of said plurality of diverse locking protocols.

1. "Uniform file-locking semantics"/"Uniform locking semantics"

Disputed Claim Term: "Uniform file-locking semantics"/"Uniform locking semantics" ('097 patent claims 1, 5, 14, 27, 31, 40, 53, 57, and 66; '351 patent claims 1, 2, 6, 9-11, 13-16, 18, 38-39, 42, 44, 47-49, 51-54, 56, 76-77, 82, 85-87, 89-92, 94, 114)	
Sun's construction	**NetApp's construction**
"A set of lock modes, each combining an access mode and a deny mode, into which lock requests from multiple protocols are translated." Sun further contends that the preambles in which this term appears are limiting.[13]	"[A] set of [file-] locking rules that can be applied in the same way to requests using diverse [file-server or file-] locking protocols."[14]

The dispute centers on whether the semantics are lock modes, which must have an access mode and deny mode as Sun contends, or whether uniform file-locking semantics are rules that are applied in the same way to diverse protocols, as NetApp contends. NetApp argues that the claims show that "uniform [file-] locking semantics" are rules, not lock modes. As NetApp notes, claim 1 provides that the uniform file-locking semantics include a list of steps for taking particular actions in response to particular requests. The rules in this claim, according to NetApp, are that the system will grant an opportunistic lock in response to a message using the first protocol, and break the lock in response to a message using the second protocol. '351 patent at Claim 1. The rules allow clients using different protocols to work together to access the same data without corrupting it. Ganger

[13] NetApp concedes that the preambles in which the term appears are limiting. NetApp also abandoned its argument that the term need not be construed at all.

[14] The bracketed portions of NetApp's construction apply to "uniform file-locking semantics" but not to "uniform locking semantics."

Decl. ¶ 45.

This claim language strongly indicates that the semantics are a set of rules composed of certain predetermined steps that mandate certain responses to certain events. While Dr. Brandt claims that these actions taken by the file server are functions or actions taken by that server, rather than a set of rules, see Supp. Brandt. Decl. ¶ 23, requiring a certain action in response to a given event is more accurately characterized as a "rule" as NetApp argues, as opposed to a function. Claim 1, for example, discusses "steps for enforcing a uniform file-locking semantics." The semantics are the rules that are "enforced" in order to mandate certain actions in response to certain events. In addition, the use of the word "enforce" in the patent title and in the patent abstract indicates that the semantics are rules. The steps outlined as the uniform locking semantics cannot be accurately described as merely a "set of modes," as Sun argues.[15]

Turning to the specification, the patent generally contemplates translating the various types of lock requests received from the file server protocols into uniform file - locks, each containing an access mode and a deny mode:

> The file server 110 uses a uniform file-locking semantics so as to model file-locking aspects of any requested operation from any file server protocol in the same way. The uniform file-locking semantics identifies a uniform set of file -locks, each including an access-mode for the requesting client device 130 and a deny-mode for all other client devices 130.

'351 patent at 7:38-44.

NetApp argues that this aspect of the preferred embodiment shows that the "uniform file-locking semantics" identifies the "uniform set of locks," but the former is not equated with the latter. Ganger Decl. ¶ 47. In addition, the uniform file-locking semantics model file-locking in the same way, i.e., uniformly, for any protocol, according to Dr. Ganger. The uniform file-locking semantics do so by providing uniform rules regarding

---

[15] NetApp also notes that U.S. Patent No. 5,668,958, cited on the face of both patents, states: "The API also comprises the rules or semantics of behavior of the file system i.e. the effects of various file system requests on objects in file system repository." Sun argues that this merits less weight than evidence of the patentee's own words. It is also is unclear whether or not the semantics are more properly categorized as rules or effects in the '958 patent. Therefore, while this reference in the '958 patent mildly supports NetApp, it is not particularly persuasive.

locking, including (1) how requested lock modes are determined for any request from a client using any particular protocol and (2) how requested lock modes are compared to existing lock modes to decide whether to allow or deny a given request. Ganger Decl. ¶ 47. The Court agrees with NetApp, as the above passage demonstrates that the locking semantics are not the same as lock modes. Rather, the semantics identify a set of file - locks, which include certain modes (access and deny modes). These semantics are a set of rules or commands that can be applied in the same way to different requested operations.

Dr. Brandt argues that the same passage shows that the uniform locking semantics establish a uniform representation of file-locks on the server, which consists of a set of lock modes, each including an access mode and deny mode. Brandt Supp. Decl. ¶¶ 20-21. This portion of the specification indicates that the set of file-locks which are identified by the uniform file-locking semantics include an access mode and a deny mode. See ’351 patent at 4:52-56 (defining “lock mode” as the combination of an access mode and a deny mode). Dr. Brandt argues that no other embodiment, other than this teaching of a uniform file-locking semantics having an access mode and a deny mode, is taught or suggested in the specification. Brandt Decl. ¶¶ 70-71. While this portion of the specification uses lock modes having an access mode and deny mode, these lock modes are not utilized in this way for every aspect of the invention.

Specifically, the invention describes three separate aspects of the invention: (1) one in which the uniform file-locking semantics protect against data corruption; (2) another one in which the common internet file system (CIFS) client device can obtain an oplock and the network file system (NFS) and network lock manager (NLM) devices[16] are allowed to request to break the oplock; and (3) a third aspect in which the CIFS client device can obtain a “change-monitoring lock.” See ’351 patent at column 2. The patents disclose an application of uniform locking semantics in which lock modes are not involved at all – the

---

[16] The background of the invention explains that in the prior art, there are multiple diverse file server protocols, each with differing semantics for file operations. The NFS protocol does not provide semantics for file-locking. The CIFS protocol has an extensive mandatory file-locking semantics. And, while NFS is often augmented by a NLM protocol, NFS treats NLM locks as advisory only.

ability of a CIFS client device to obtain a "change-monitoring" lock on a file directory. This type of lock gives notice when a directory is changed by CIFS or non-CIFS devices. '351 patent at 2:51-59. For example, this "change-monitoring" lock provides notice when a directory is created, deleted, or files are renamed or moved. Id. Such a lock is not exclusive, does not prevent access to a file, and cannot properly be explained in terms of access modes and deny modes. Ganger Decl. ¶ 49. Lock modes are not even mentioned in the description of change-monitoring locks in the specification. '351 patent at 14:20-15:6 (noting that the change monitoring lock specifies both the name of the changed file and the type of change). Yet "uniform file -locking semantics" are used when granting change-monitoring locks. See, e.g., claims 6 and 85. While Sun maintains that the access and deny lock modes constitute the uniform language of these patents, the fact that the change monitoring lock cannot be expressed in mere access and deny modes severely undermines Sun's proposed construction.

Sun counters that lock modes are involved in granting a change-monitoring lock, as that process involves receiving a file open request. '351 patent at 14:30-35. In order to determine whether such a request should be allowed, the CIFS open request must be translated into a uniform lock mode, containing an access mode and deny mode. Id. at 9:37-10:33; Brandt Supp. Decl. ¶¶ 18-19. However, while lock modes may be used in granting a change-monitoring lock, the patent does not address what lock mode exists once the change-monitoring lock is in effect. Sun concedes that the patent is silent on this point, but argues that the mode would likely be a deny-none mode. The patent does not support Sun's conclusion, however, and a deny-none mode itself would not provide change notification to a client. Furthermore, the specification notes that the "file-lock" is converted on the open directory to a change-monitoring lock. '351 patent at 14:38-39. When the change-monitoring lock is in effect, therefore, there is no specified lock mode.

Sun's reliance on Respironics, Inc. v. Invacare Corp., 303 Fed. Appx. 865, 871 (Fed. Cir. Dec. 16, 2008) is inapposite, as in that unpublished case, the patent had only one embodiment, in which the pressure magnitudes at issue were "predetermined." Limitation

of the claim term at issue in Respironics was supported by the patent's only embodiment. Here, by contrast, one of the three aspects of the invention – the change-monitoring lock – does not appear to utilize the uniform lock modes proposed by Sun when carrying out its change notifying function.

As to NetApp's proposed construction, Sun argues that it is flawed because the specification does not teach that the uniform file-locking semantics are applied in the same way to requests using different protocols. In particular, the patent specification describes how the uniform file-locking rules are applied differently to requests from different CIFS, NFS, and NLM protocols. Brandt Decl. ¶¶ 74-75; '351 patent at columns 6-13 (outlining differences between application of the rules for CIFS, NFS, and NLM requests). For example, NLM and CIFS byte-range lock requests are handled differently. With respect to NLM, "[i]f the file server 110 is checking for conflicts between existing file -locks or byte range locks, and a new request for a NLM byte-range locks, the . . . locks are cross-indexed against a lock mode equivalent to the new NLM byte-range lock request. For the purpose of comparing with existing file -locks, the file server 100 treats newly requested NLM byte-range locks as having deny-mode deny-none, and as having access-mode read-only for nonexclusive locks . . . and access-mode read-write for exclusive locks." By contrast, "CIFS byte-range lock requests are only checked against byte-range locks because they require a prior CIFS file open operation at which existing file -locks were already checked." Brandt Decl. ¶ 75 (quoting '351 patent at 11:32-38, 12:25-28).

NetApp counters that Sun fails to distinguish between: (1) the locking rules, which apply uniformly to locking requests by each client, no matter what protocol that client uses; and (2) the variation among clients as to which type of locking request a particular client might make. The background of the invention itself notes "that it is desirable to provide a method and system for enforcing file-locking semantics among client devices using multiple diverse file server protocols," which is achieved in the invention by using a "uniform set of file-locking semantics." '351 patent at 1:60-67. In the preferred embodiment, specific file-locking semantics of the CIFS protocol are implemented to allow

NFS client devices to inter-operate with CIFS client devices. Id. at 2:1-5. And, as noted above, the preferred embodiment states that the file server "uses a uniform file-locking semantics, so as to model file-locking aspects of any requested operation from any file server protocol **in the same way**." Id. at 7:38-44 (emphasis added). For example, in a system providing for opportunistic locks, the uniform locking semantics are applied in the same way to all requests for opportunistic locks. However, not all clients may be able to request opportunistic locks. Ganger Decl. ¶ 56. As the summary of the invention describes the invention: "uniform file-locking semantics provides that the file server determines, before allowing any client device to read or write data, or to obtain a new file -lock or byte-range lock, whether that would be inconsistent with existing locks, regardless of originating client device and regardless of originating file server protocol or file-locking protocol for those existing locks." '351 patent at 2:21-27.

As NetApp itself acknowledges, "clients using diverse protocols may make different types of requests and therefore trigger different subsets of the rules." NetApp Markman Hearing Slide 93. In other words, the same subset of "rules" do not apply to all requests from all protocols. In addition, the initial translating steps are different for different protocols. For example, both CIFS and NFS protocols issue read requests. For CIFS read requests, the lock mode of the request is compared only against the access mode acquired when the file was opened, after file open time. '351 patent at 4:35-45. On the other hand, for NFS read requests, the lock mode of the request is compared against the lock mode of any pre-existing file or byte-range locks. Id. at 6:27-30, 6:46-50, 11:26-30. NetApp's proposed construction, which states that the rules are applied "in the same way," is therefore somewhat misleading. In addition, the semantics do more than just translate the requests, as they also compare requests. See Sun Reply Brief at 10:18-22 (citing '351 patent at 9:37-10:33 (describing method by which CIFS open request is translated into a uniform lock mode and the requested lock mode is compared with the mode of any preexisting locks to determine whether file access may be granted). In other words, the rules govern file-locking aspects of the request, rather than just translate those requests.

In light of the above, the Court proposes construing "uniform file -locking semantics/uniform locking semantics" as "[a] set of [file] locking rules that consistently govern [file]locking aspects of any request, even though the requests use diverse locking protocols." While this is a proposed construction, the parties may only comment if they find a mistake or ambiguity in the wording, as opposed to disagreeing with the Court's reasoning, and they must do so within ten days of the date of this Order.

2. "Opportunistic Locks"

Disputed Claim Term: "Opportunistic Locks" ('097 patent claims 1, 14, 24, 27, 40, 50, 53, 66, and 76; '351 patent claims 1, 2, 11, 29, 38-39, 42, 49, 67, 76-77, 87, 105, and 114)	
Sun's construction	**NetApp's construction**
Exclusive access to a file which can be revoked by attempted access by another client.	No construction necessary, plain and ordinary meaning. To the extent the Court deems a construction necessary, "exclusive lock that is given to only one client at a time, and that permits that only one client to read or write the locked data until another client device attempts to read or write that data."

At the hearing, the Court noted that the parties did not seem to have a substantive dispute regarding this claim term. After further conferring, the parties agreed to the following construction of the term, which the Court adopts: "Exclusive access to a file or a portion of a file allowing one or more reads or writes, that is given to only one client at a time, which can be broken by another client's attempt to access the file or a portion of a file."

IV. CONCLUSION

In accordance with the foregoing, and for the reasons discussed above, the Court construes the disputed terms of the parties' patents as follows:

1. "Token" is "a data structure consisting of a name; an identifier of an element with which the token is associated; an identifier of the method by which the value field may be populated; and the value field, which holds the value associated with the element, which can include an empty value." While this is a proposed construction, the

parties may only comment if they find a mistake or ambiguity in the wording, as opposed to disagreeing with the Court's reasoning, and they must do so within ten days of the date of this Order.

2. "Computer system" is "a system that includes at least one computer and that may contain a number of computers coupled in a network. Even where a computer system contains only one computer, that computer must be able to communicate via email or a modem connection or another communication channel to allow remote monitoring."

3. "Modifiable Tree Structure Including Elements in A Fixed Hierarchical Relationship"/"Modifiable Static Tree Structure" is "a tree structure comprising elements, which does not vary according to the system being monitored, and which can be changed to add or delete elements representing hardware or software components on a computer system."

4. "To examine [examining] the event communication to determine whether the event is associated with at least one object or object level on a selected object list having at least one specified object or object level" is "to evaluate [evaluating] the event information received from a network device to determine if the event information relates to any item on a selected list of one or more network devices or network device levels."

5. "Discovery interface" is "one or more software modules that receive information from a fabric driver and provide the information to an application."

6. "In a non-fixed pattern" is "in such a way that the redundant information can be placed within the stripe on unallocated blocks in any order."

7. "Initiator group (igroup)" is "a logical named entity that may have a human friendly name assigned to one or more addresses associated with one or more initiators. As used in the patent, 'human friendly name' means an arbitrary label selected by the user or administrator that may be a spoken name, a path designation or include a hierarchical naming convention."

8. "Uniform [file-]locking semantics" is "[a] set of [file] locking rules that consistently govern [file]locking aspects of any request, even though the requests use

diverse locking protocols." While this is a proposed construction, the parties may only comment if they find a mistake or ambiguity in the wording, as opposed to disagreeing with the Court's reasoning, and they must do so within ten days of the date of this Order.

9. "Opportunistic Locks" is "exclusive access to a file or a portion of a file allowing one or more reads or writes, that is given to only one client at a time, which can be broken by another client's attempt to access the file or a portion of a file."

IT IS FURTHER ORDERED that a case management conference is set for **July 8, 2009 at 2:00 p.m**, to discuss setting further dates in the case. The parties shall file a joint case management statement no later than **July 1, 2009.** The parties shall propose a schedule for filing summary judgment motions and address whether to limit the number of such motions and whether to stagger and prioritize the motions.

IT IS SO ORDERED.

Dated: May 29, 2009

Elizabeth D. Laporte

ELIZABETH D. LAPORTE
United States Magistrate Judge